UNIVERSIDAD CARLOS III DE MADRID

“DESARROLLO DE UNA APLICACIÓN PARA DISPOSITIVOS MÓVILES EN ANDROID”

Trabajo de Fin de Grado

Grado en Ingeniería Electrónica Industrial y Automática

Autor: Sergio Sánchez Fernández

Tutora: Concepción Alicia Monje Micharet

AGRADECIMIENTOS

En primer lugar me gustaría dar las gracias a mi tutora, Concepción Alicia Monje, por el tiempo dedicado y su ayuda a lo largo del desarrollo del proyecto. También me gustaría agradecer la colaboración desinteresada de Jorge García Bueno durante el desarrollo de la aplicación, ya que sin su ayuda no hubiera sido posible su realización.

Dar las gracias a mis padres José Miguel y Pilar por educarme, aguantarme y por convertirme en la persona que soy hoy en día. Desde que aprendí a escribir mi primera palabra hasta la última letra que aparece en esta memoria han sido gracias a su apoyo y dedicación.

A mi hermano Miguel Ángel por ser un gran apoyo durante toda la carrera, por ayudarme en todo lo que estuviese en su mano y un poco más y por tener siempre palabras de ánimo en cualquier momento. Gracias también a su señora esposa Vicky, por sus ánimos y por saber siempre ver el lado positivo de las cosas.

Gracias también a mis amigos de toda la vida, Alberto, Alex, Laura, Manu, Pedro y Víctor por estar siempre a mi lado acompañándome en esta aventura que es la vida, tanto en los buenos como en los malos momentos.

A mis compañeros de universidad, por ayudarme a tomarme con filosofía los momentos duros y de estrés durante los años de carrera, y por hacer agradable cada minuto pasado en el campus. Gracias por todo ello a Luis Carlos, Davicito, Rober, Fran, Ricky, David, Suquillo, Roberto, Nacho, Álvaro y Chen.

RESUMEN DEL PROYECTO

En el siguiente documento se va a explicar paso por paso el proceso seguido para conseguir realizar una aplicación para dispositivos móviles con sistema operativo Android.

La aplicación que se ha desarrollado consiste en una galería de imágenes creada por la pintora Marina Anaya. Estas imágenes se encuentran asociadas a un poema o frase, escrito por la periodista Lidia Martín. El usuario al pulsar encima de una de ellas provocará que esta se gire y muestre su texto correspondiente.

A lo largo del proyecto se mostrarán las pautas que se han seguido para lograr el funcionamiento eficiente de la aplicación, desde la instalación del programa necesario para realizar la programación, hasta la prueba final en un dispositivo móvil, pasando por las diferentes fases del diseño, que se explicarán detalladamente.

Finalmente se explicaran los pasos para incluir la aplicación en la Play Store, que es la página donde se almacenan todas las aplicaciones disponibles para Android.

ABSTRACT

The following document will explain step by step the process followed for making an application for mobile devices with Android operating system.

The application that has been developed is a gallery of images created by the artist Marina Anaya. These images are associated with a poem or phrase, written by the journalist Lidia Martín. When the user clicks an image it turns and displays the corresponding text.

Throughout the Project will show the guidelines that have been followed to ensure the efficient operation of the application, from installing the required software for programming, to final testing on a mobile Device through the different spelled design phases.

Finally we will explain the steps to include the application in the Play Store, which is the page where all the apps available for Android are store.

INDICE

Tabla de contenido

INDICE DE FIGURAS

1. INTRODUCCIÓN

1.1 SMARTPHONES Y SISTEMAS OPERATIVOS

La telefonía ha evolucionado a pasos agigantados los últimos años. Hace poco más de una década nadie usaba móviles. En la actualidad, se hace imposible la idea de salir de casa sin nuestro dispositivo en el bolsillo. El poder estar conectados a internet casi desde cualquier sitio, sumado a una extensa variedad de aplicaciones, han provocado el aumento exponencial del mercado de esta tecnología.

1.1.1 SMARTPHONES

Un Smartphone es un teléfono móvil que tiene unas características diferentes, muchas de ellas propias de los ordenadores personales. Los primeros modelos ya conocidos como Smartphone surgieron en 1993, cuando IBM sacó al mercado un modelo llamado "Simon", cuya imagen se muestra en la figura 1. Este teléfono tenía características como calendario, bloc de notas, calculadora y otras aplicaciones sencillas. Su peso, dimensiones y sobre todo precio lo convertían en una tecnología al alcance de pocos.

Figura 1. Teléfono móvil Simon de IBM

Durante varios años surgieron nuevos terminales (ver Figura 2), pero no fue hasta la salida del iPhone de Apple cuando surgió la revolución Smartphone en el mercado con móviles potentes con tecnología punta y con precios al alcance del usuario medio.

Figura 2. Evolución tecnológica de los teléfonos móviles.

Desde entonces los móviles han ido evolucionando y cada vez son más las posibilidades que ofrecen por medio de aplicaciones tanto gratuitas como de pago. Por ello, funciones como GPS, juegos, reproducción de audio y video, correo electrónico y gestión de redes sociales son habituales en cualquier Smartphone.

Hoy en día la inmensa mayoría de estos teléfonos son táctiles, lo que permite al usuario una interacción más rápida con el dispositivo. El mercado de los Smartphones ya superó en 2011 en ventas al mercado de los ordenadores y sigue así hasta hoy en día debido a que los desarrolladores prefieren invertir en productos para móvil, ya que encuentran más beneficio que en las computadoras.

1.1.2 SISTEMAS OPERATIVOS PARA MÓVILES

1.1.2.1 ANDROID

Android nace en 2005 y es un sistema operativo cuya base es Linux y fue diseñado para ser capaz de adaptarse tanto a todo tipo de dispositivos móviles como al emergente mercado de tabletas digitales. Al comienzo la compañía nació bajo el nombre Android Inc. y fue en julio de 2005 cuando Google adquirió la misma. Ese mismo año se creó la característica mascota de Android, Andy (Figura 3). Esta compra responde a la intención por parte de Google de hacer su entrada en el mercado de los teléfonos móviles.

Figura 3. Andy, la mascota de Android.

Desde su primera aparición en una versión definitiva en septiembre de 2008 bajo el nombre Android 1.0, el sistema operativo ha ido sufriendo actualizaciones y mejoras para adaptarse al avance tecnológico de los móviles. Como curiosidad, comentar que estas actualizaciones reciben un nombre, por orden alfabético, relacionado con postres. Así pues, las sucesivas actualizaciones de Android han recibido los siguientes nombres: Apple Pie, Banana Bread, Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich y Jelly Bean para la última, que corresponde a la versión 4.2 que apareció en el mercado el 13 de Noviembre de 2012.

Actualmente Android es líder del mercado frente a sus competidores, ya que su sistema operativo está incluido en más del 70% de los dispositivos móviles que se venden (Figura 4).

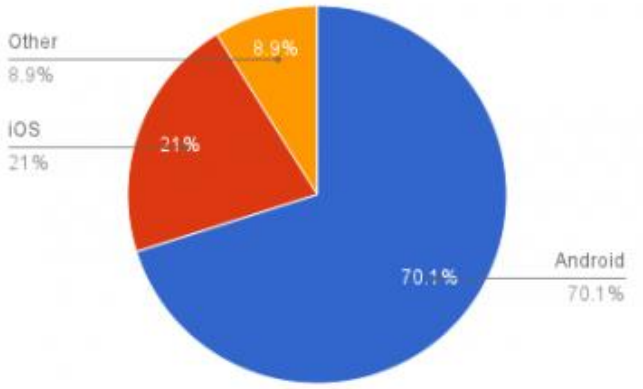


Figura 4. Estadísticas ventas de móviles finales de 2012.

Sin embargo, este dominio se pierde cuando hablamos del mercado correspondiente a las Tablets, ya que en este apartado Apple, con su sistema operativo iOS, es líder de ventas (Figura 5).

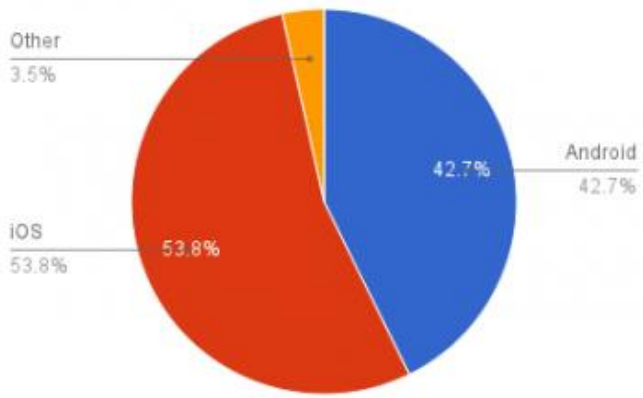


Figura 5. Estadísticas de ventas de Tablets finales de 2012.

Al contrario que otros sistemas operativos, Android se desarrolla de forma abierta, eso lo convierten en una ventaja para desarrolladores y para los usuarios. Múltiples aplicaciones permiten modificar los móviles y sus funciones de manera sencilla y efectiva [1].

1.1.2.2 OTROS SISTEMAS OPERATIVOS

IOS

Perteneciendo a la empresa Apple, apareció por primera vez en Enero de 2007 como sistema operativo para el iPhone (Figura 6), primer teléfono móvil de la compañía y que fue una revolución en el mercado. Una revolucionaria interfaz multitáctil llena de detalles, convirtió al iPhone y su sistema operativo en líder del mercado y marcaba las pautas del futuro tecnológico de los móviles.

El sistema operativo iOS ha ido evolucionando a la par que Apple iba sacando un nuevo modelo de iPhone al mercado. El diseño, su facilidad de uso y una variedad de aplicaciones y juegos enorme han convertido a iOS en un referente. Sin embargo, el sistema operativo de Apple es cerrado y por lo tanto existe menos libertad a la hora de modificar el teléfono. Esto, unido al elevado precio de los productos de Apple, ha provocado que Android esté hoy por delante en el mercado.

Figura 6. IPhone 5.

WINDOWS PHONE

Este sistema operativo está desarrollado por Microsoft y es el sucesor de la plataforma Windows Mobile. Su presencia en el mercado es relativamente joven, ya que hizo su aparición en el año 2011 cuando Microsoft llegó a un acuerdo con Nokia para utilizar su sistema operativo en todos los móviles de este fabricante. Su diseño (Figura 7) está pensado para resultar similar a las versiones de escritorio de Windows. La entrada en mercado de este sistema operativo ha sido bastante decepcionante y Microsoft sigue desarrollando nuevas funciones y precios más baratos para poder competir cara a cara tanto con Android como con iOS.

Figura 7. Interfaz de Windows Phone.

BLACBERRY OS

Es el sistema operativo usado para los dispositivos BlackBerry (Figura 8). Es el cuarto más utilizado pero su falta de actividad ha reducido su cuota de mercado y su futuro se presenta complicado. Su orientación es para uso profesional como gestor de correo electrónico y agenda.

Figura 8. Interfaz BlackBerry OS.

FIREFOX OS

Recibe el nombre del famoso navegador de internet y ha surgido con el apoyo de grandes compañías fabricantes como Alcatel o Sony. Su uso principal es en móviles de gama baja. Tiene en su haber un amplio grupo de desarrolladores creando aplicaciones pero la falta de un móvil potente que lo incluya limita su presencia en el mercado.

UBUNTU

Todavía está en fase de desarrollo y llegará en 2014 al mercado. Está basado en Android con unos ligeros cambios y se adapta a cualquier tipo de móvil tanto de gama baja como de alta.

1.1.3 LA PLATAFORMA ANDROID

1.1.3.1 ECLIPSE

Eclipse es el Entorno íntegro de desarrollo (Integrated Development Environment – IDE) que utilizaremos para programar nuestra aplicación. El lenguaje de programación que usa es Java. Eclipse se entiende como un marco y un conjunto de servicios para construir un entorno de desarrollo a partir de componentes conectados (plug-in). Existen plug-ins para el desarrollo de Java (JDT Java Development Tools) así como para el desarrollo de C/C++, COBOL, etc. Fue desarrollado por IBM.

1.1.3.2 JAVA

Java es el lenguaje de programación orientado a objetos en el que se basan las aplicaciones para Android. Fue desarrollado en sus comienzos por James Gosling e hizo su aparición en 1995. Cuando hablamos de un lenguaje orientado a objetos nos referimos a un tipo de programación en el que los distintos tipos de datos que se usen estén unidos en sus operaciones. Está basado en varias técnicas, incluyendo herencia, modularidad, polimorfismo y encapsulamiento.

1.1.3.3 SDK ANDROID

Para poder iniciar el desarrollo de la aplicación necesitaremos el Software Development Kit (SDK) de Android. El SDK contiene todas las librerías y atributos necesarios para desarrollar en Android, así como poder probar aquellas aplicaciones en las que estemos trabajando.

El SDK incluye un conjunto de herramientas entre las que están un depurador de código, un simulador de teléfono, biblioteca, documentación y ejemplos de código.

Según avanzan las nuevas versiones de Android, las actualizaciones del SDK van saliendo, pero siempre soportan versiones antiguas para tener la opción de desarrollar aplicaciones para móviles menos actuales [2].

1.2 OBJETIVOS

El objetivo fundamental de este proyecto es la creación de una aplicación para cualquier dispositivo móvil que funcione con el sistema operativo Android. La aplicación consistirá en un carrusel de veinte imágenes, que tienen asociadas cada una de ellas un texto único. El usuario al pulsar en cualquier parte de una de las imágenes provocará que esta gire y muestre su texto. Para que la aplicación final se ajuste a lo buscado se seguirán una serie de objetivos:

- La aplicación debe ser accesible al mayor número de terminales posibles, siendo consciente de que es una aplicación de funcionamiento táctil.
- Debe ser fácil e intuitiva, dando prioridad visual a las imágenes.
- Cada texto debe estar asociado a su imagen correspondiente. No deben aparecer nunca repetidos o mezclados.
- Incluir dentro de la aplicación información sobre la autora, tanto de los dibujos como de los textos.
- La aplicación debe de ocupar el menor número posible de memoria.

1.3 ESTRUCTURA DEL DOCUMENTO

El siguiente documento se encuentra dividido en diferentes capítulos según se ha ido avanzando en el desarrollo de la aplicación. A continuación se muestra un breve resumen sobre el contenido de cada uno de los capítulos:

- Capítulo 1: Introducción al mundo de los Smartphones y concretamente al sistema operativo Android, hablando sobre los principales elementos utilizados en el desarrollo de aplicaciones.

- Capítulo 2: En este capítulo se explicara el proceso llevado a cabo para instalar todo lo necesario para empezar a programar y crear la aplicación. Además se realiza una primera aplicación sencilla de ejemplo y se incluye información sobre el ciclo normal de cualquier aplicación en Android.

- Capítulo 3: Se describe el proceso llevado a cabo para realizar la aplicación por completo. Se muestran los detalles más importantes y relevantes y los puntos de mayor complicación durante el desarrollo.

- Capítulo 4: Se explica cómo subir una aplicación a Google Play para poder hacerla accesible a todos los usuarios de este sistema operativo.

- Capítulo 5: Conclusiones después de realizar la aplicación y posibles mejoras a incluir en el futuro.

- Capítulo 6: En este capítulo se hace referencia al presupuesto total necesario para llevar a cabo el desarrollo de la aplicación.

- Capítulo 7: Mención y referencia a libros, artículos y páginas web usadas durante la elaboración y el diseño de la aplicación.

2. INSTALACIÓN DEL ENTORNO DE DESARROLLO

A lo largo de este capítulo se van a explicar los pasos que se han de seguir para poder instalar el entorno de desarrollo de Android. Desde la instalación de Eclipse, que es el programa usado para la realización de la aplicación, hasta la creación de la primera aplicación. Además se detallarán los principales componentes de un proyecto generado en Eclipse y una explicación sobre el funcionamiento normal de una aplicación y sus componentes.

2.1 INSTALACIÓN DE ECLIPSE Y EL SDK DE ANDROID

Lo primero y fundamental es la instalación del programa Eclipse, totalmente gratuito y que la propia página de Android para desarrolladores ofrece para descargar directamente [3]. Junto con Eclipse, la propia descarga incluye el SDK de Android.

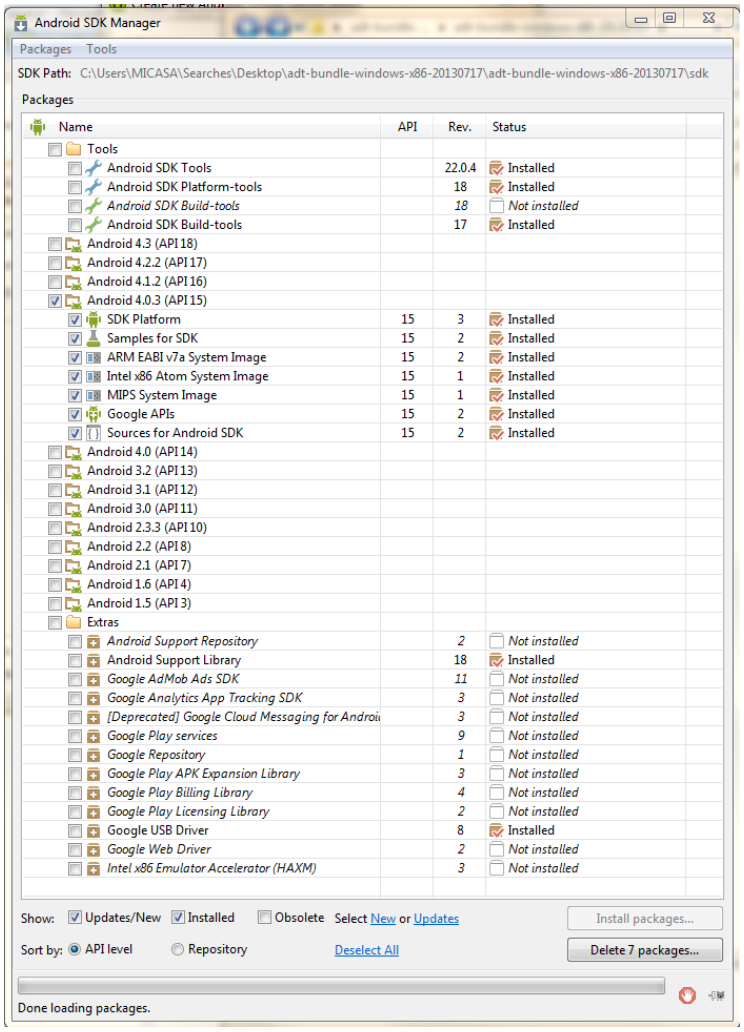


Figura 9. Android SDK Manager.

Una vez descomprimido el archivo, nos encontramos una carpeta con el programa Eclipse listo para ejecutarlo. En otra carpeta nos encontraremos un ejecutable llamado SDK Manager que nos permitirá descargar el SDK para eclipse. Si lo abrimos, vemos las diferentes versiones que ha habido de Android y la opción de descargarlas para poder trabajar en ellas (Figura 9). Lo recomendable en estos casos es instalar no la última versión, sino una intermedia. Esto es debido a que las APIs más modernas sólo funcionarán en los móviles más actuales y eso restará el volumen de móviles en los que nuestra aplicación funcionará correctamente. Utilizaremos la versión 4.0.3 que es relativamente nueva.

Ya descargados e instalados los paquetes, el siguiente paso será crear un dispositivo para poder simular las aplicaciones.

2.2 CREACIÓN DE UN SIMULADOR VIRTUAL MÓVIL

Los Android Virtual Devices (AVD) son unas herramientas fundamentales para los desarrolladores, ya que permiten emular en nuestro ordenador un dispositivo móvil para poder probar nuestra aplicación, sin necesidad de instalarla en el móvil. En eclipse podremos acceder a esta opción en *"Windows/Android Virtual Device Manager"*. Dentro le daremos a "New" y se nos abrirá ventana de la Figura 10.

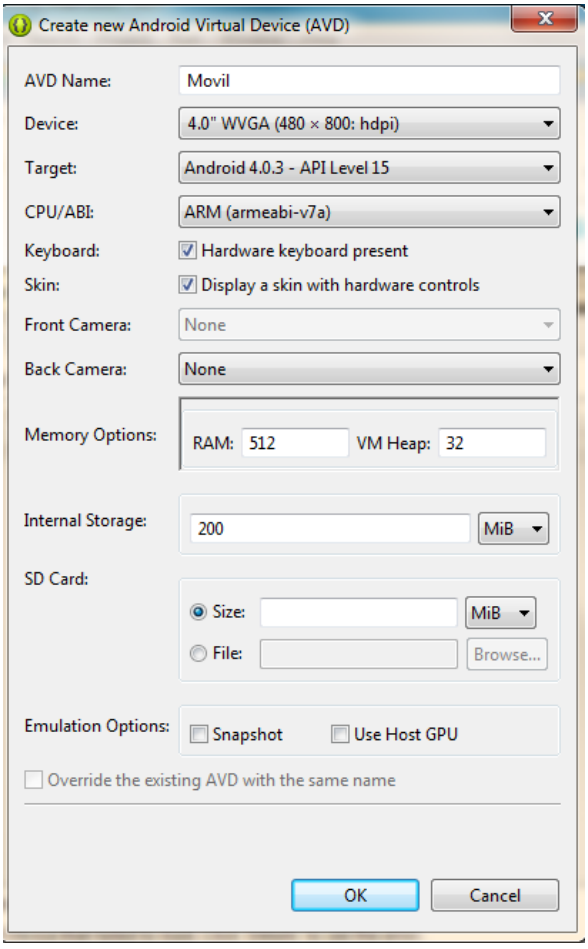


Figura 10. Creación de un nuevo AVD.

Tendremos varias opciones, como por ejemplo poner el nombre al dispositivo que utilicemos. Después tendremos que elegir el móvil que emulará el ordenador. Existen varias opciones con diferentes tamaños de pantalla según lo que pretendamos. Elegiremos una pantalla por defecto de cuatro pulgadas, ya que hoy en día la mayoría de los móviles del mercado rondan ese tamaño. El Target indica la versión de Android que correrá sobre nuestro dispositivo. Estas son las principales opciones, el resto son detalles para personalizarlo más en el caso de que queramos una simulación más específica.

Una vez hecho esto, ya podemos ejecutar nuestro AVD obteniendo como resultado la simulación de un móvil con varias aplicaciones y opciones por defecto como calendario, calculadora, reloj, etc. [4]

2.3 SIMULACIÓN DE LA PRIMERA APLICACIÓN

Una vez tenemos todo lo necesario, podemos ejecutar nuestra primera aplicación Android en nuestro dispositivo virtual. Para esta prueba utilizaremos la que se genera por defecto en Eclipse y que no es otra que el conocido "Hello World!". Deberemos ir a *"File/New/Android Application Project"*. Se nos abrirá una venta con distintas opciones (Figura 11):

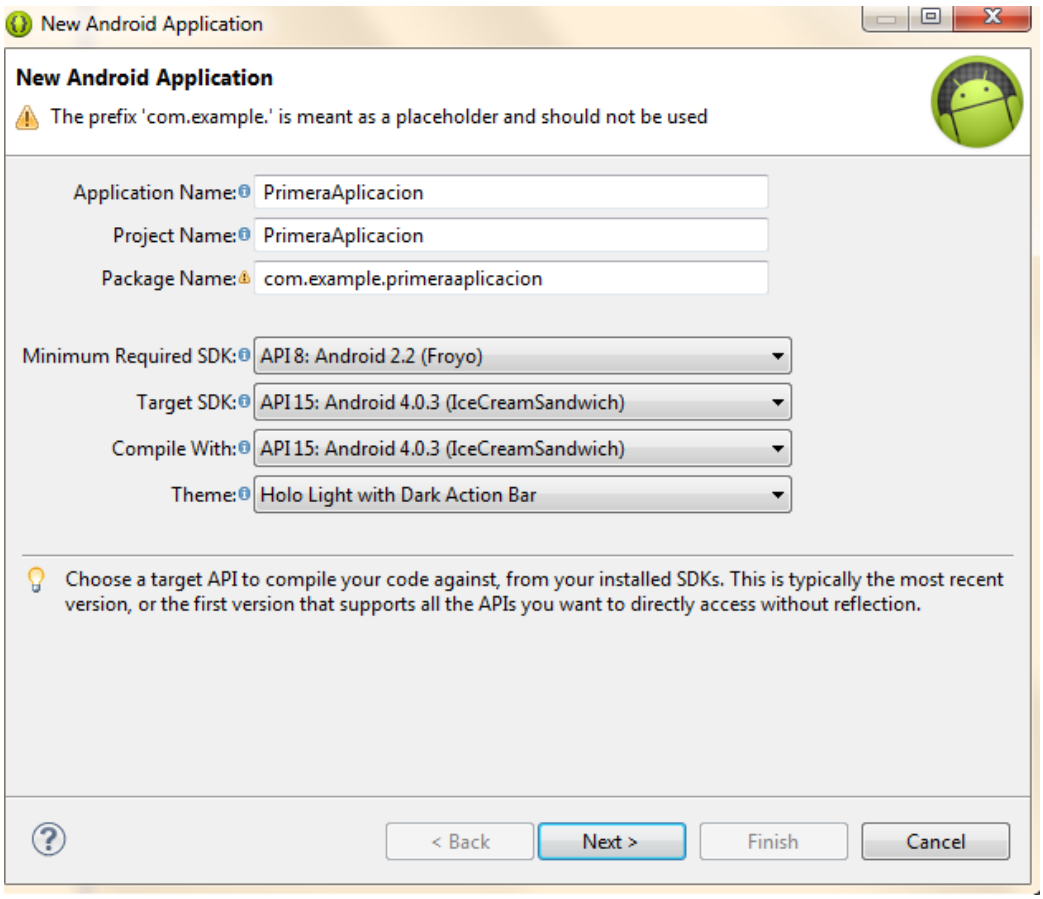


Figura 11. Creación de una nueva aplicación de Android.

- **Application Name:** El nombre de nuestra aplicación.
- **Project Name:** El nombre de nuestro proyecto (por defecto el mismo que la aplicación).
- **Package Name:** Es el nombre del paquete al que estará asociado la aplicación.
- **Minimum Required SDK:** Indica el nivel mínimo de API que requiere la aplicación para funcionar correctamente. Por debajo de este no funcionará correctamente.
- **Target SDK:** Es la versión de SDK para la cual estamos desarrollando la aplicación. Se recomiendo usar para esto la última versión estable del SDK.
- **Compile With:** Este campo hace referencia al SDK que utilizaremos para compilar el proyecto.
- **Theme:** Hace referencia a los estilos que definen la apariencia de la aplicación. Utilizaremos el valor por defecto. En un futuro puede modificarse por parte del usuario.

Una vez hecho esto, nos aparecen otras opciones como el fondo o la imagen de la aplicación. Aceptamos las que vienen por defecto, ya que esto también podremos modificarlo más adelante. Con todos los campos completados, se nos crea automáticamente el proyecto con la aplicación "Hello World!". Para simularla en nuestro dispositivo virtual, pulsaremos con el botón derecho sobre el proyecto y seleccionaremos *"Run as/ Android Application"*.

Esto provocará que se nos abra el ADV y nos muestre la aplicación en la pantalla (Figura 12).

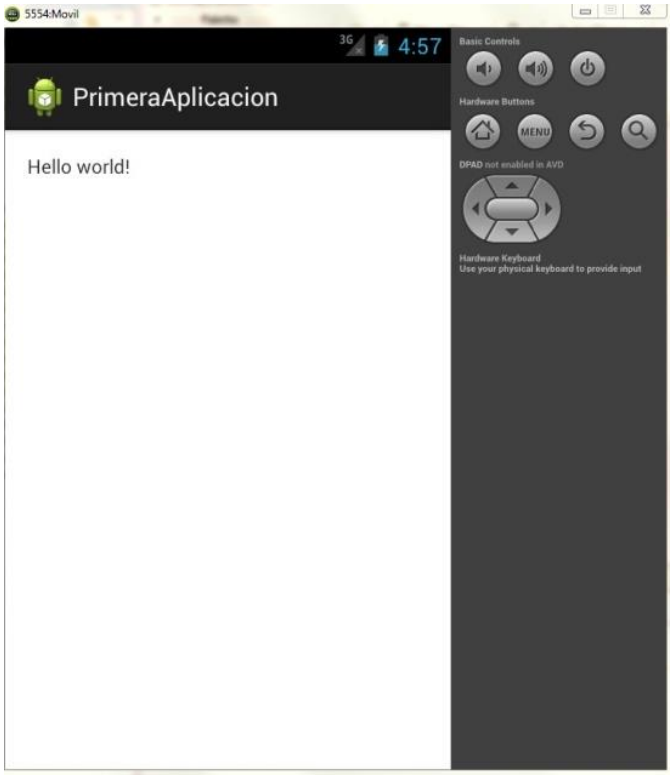


Figura 12. Primera aplicación ejecutada en el ADV.

Como se ve en la Figura 12, el ADV incluye botones para poder navegar e interactuar con la aplicación. Además, arrastrando el ratón por la pantalla y pulsando en ella, el dispositivo reaccionará, pues la pantalla está definida como táctil.

En la parte de arriba nos aparece el nombre que le hemos dado a nuestra aplicación ("PrimeraAplicación") junto con el logotipo, que en nuestro caso es el predeterminado de Android. Si pulsamos el botón *"Home"*, el ADV nos volverá a la pantalla inicial del móvil donde veremos el icono de nuestra aplicación y el resto de nuestros proyectos, en el caso de que tuviéramos alguno más (Figura 13).

Figura 13. Pantalla de inicio del ADV.

2.4 ESTRUCTURA EN ECLIPSE DE UN PROYECTO ANDROID

Cuando abrimos un nuevo proyecto de Android en Eclipse, automáticamente se generan una serie de carpetas y archivos que son necesarios para que tanto las simulaciones como su funcionamiento en un dispositivo móvil ocurran de manera correcta. A continuación detallaremos las partes más importantes (Figura 14):

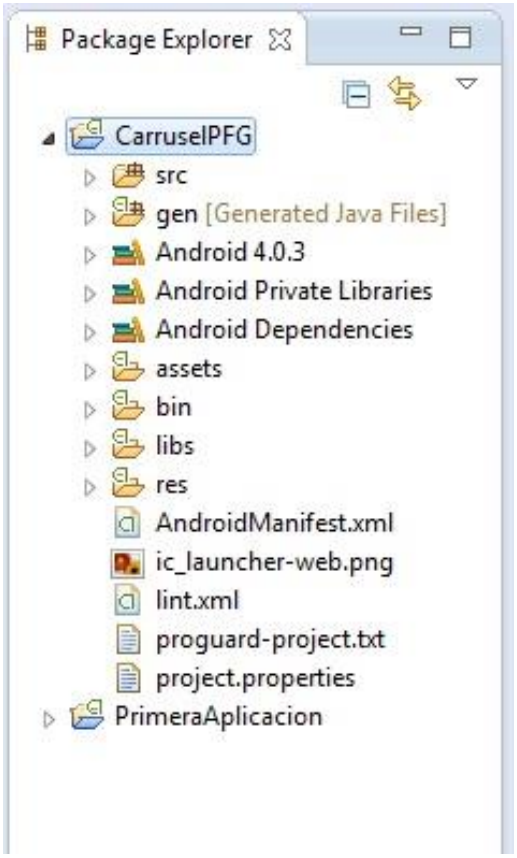


Figura 14. Estructura de un proyecto en Eclipse.

Carpeta "src": Es la carpeta más importante del proyecto. Aquí se encuentran todos los archivos de código de fuente Java que componen la aplicación, es decir, la actividad principal y el resto de clases que vayamos creando.

Carpeta "gen": Contiene clases Java que el propio programa, a través del SDK, crea de manera automática para un óptimo funcionamiento de las aplicaciones. Estos archivos no se deben modificar de forma manual. Dentro encontraremos el *BuildConfig.java* que define la constante DEBUG para que desde Java puedas saber si tu aplicación está en fase de desarrollo. Por su parte el *R.java* define una clase que asocia los recursos de la aplicación con identificadores. De esta forma los recursos podrán ser accedidos desde Java.

Carpeta "Android A.B.C": La A, la B y la C hacen referencia a la versión del SDK de Android que se esté empleando en la aplicación. En nuestro caso la 4.0.3.

Carpeta "Android Private Libraries": Son librerías necesarias para el correcto funcionamiento de la aplicación. Hasta hace poco esta carpeta se encontraba dentro de *"Android Dependencies"*, pero en la última versión se puso de manera independiente.

Carpeta "Android Dependencies": Librerías asociadas al proyecto para poder trabajar con la versión del SDK de Android que hayamos elegido para nuestro proyecto.

Carpeta "assets": En esta carpeta vamos a guardar ficheros arbitrarios que vayamos a usar en nuestra aplicación, pero a los que no queramos que se les asigne un identificador. Por ejemplo, en esta carpeta se alojan las fuentes.

Carpeta "bin": En esta carpeta es donde se compila el código y donde también se genera el archivo .apk, que es el fichero comprimido que contiene la aplicación final preparada para ser instalada.

Carpeta "libs": Sirve para contener las librerías que vamos a enlazar a nuestro proyecto. Contendrá todo los JAR (Java Archives) que se usarán durante el proyecto

Carpeta "res": Esta carpeta se denomina así por contener los recursos (resources) de nuestra aplicación, así pues, en el fichero R.java se generará y estará referenciado de manera automática todo lo que hayamos incluido en esta carpeta. Dentro de res encontramos más carpetas que tienen una nomenclatura concreta y por lo tanto, si queremos incluir alguna, deberemos conocerla. A continuación enumeramos las más importantes (Figura 15):

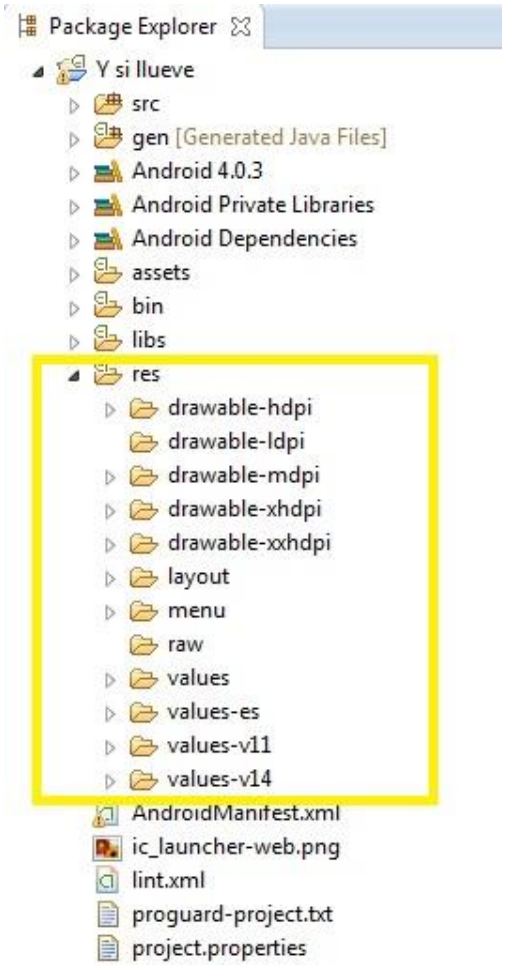


Figura 15. Carpeta "res" desplegada.

- **Drawable-*:** Estas carpetas vienen acompañadas además de por este nombre, por unas siglas (hdpi, ldpi, mdpi, xhdpi, xxhdpi) que hacen referencia a la densidad de pantalla en la que se está utilizando la aplicación.
- **Layout:** Contiene un fichero XML por cada una de las vistas de la aplicación. El formato usado es similar al HTML que se usa en el diseño de las páginas web.

- **Menu:** En esta carpeta es donde incluimos los menús de cada actividad.
- **Values:** También usa ficheros XML y su función es indicar los valores del tipo string (cadena de texto), color o estilo.
- **Anim:** No viene por defecto, pero es usada para incluir ficheros XML con animaciones.
- **Xml:** El resto de ficheros XML que sean necesarios para la aplicación
- **Raw:** Ficheros adicionales que no se encuentran en formato XML.

AndroidManifest.xml: Éste se podría entender como el archivo principal. Es una descripción de la aplicación de Android. En él se indican las actividades, intenciones, servicios y proveedores de contenido de la aplicación. También los permisos que requiere la aplicación, la versión mínima de Android y la versión sobre la que corre la aplicación.

Proguard-project.txt: Es un fichero de configuración de la herramienta ProGuard, que te permite optimizar y reducir el código generado, además de ofuscarlo para que su copia se complique.

Project.properties: Se genera automáticamente y no se debe modificar. Contiene información sobre la versión API y otras características cuando se instala la aplicación en el terminal.

2.5 DISEÑO A TRAVÉS DE GRAPHICAL LAYOUT

Eclipse ofrece la posibilidad de diseñar la parte visual de nuestra aplicación mediante herramientas gráficas. Esto facilita mucho la programación de los menús, ya que le permite al usuario hacerlo de una manera fácil e intuitiva. La opción para poder programar usando el Graphical Layout se encuentra en la parte inferior izquierda. Al lado encontramos el botón que nos permite programar escribiendo el código directamente (Figura 16).

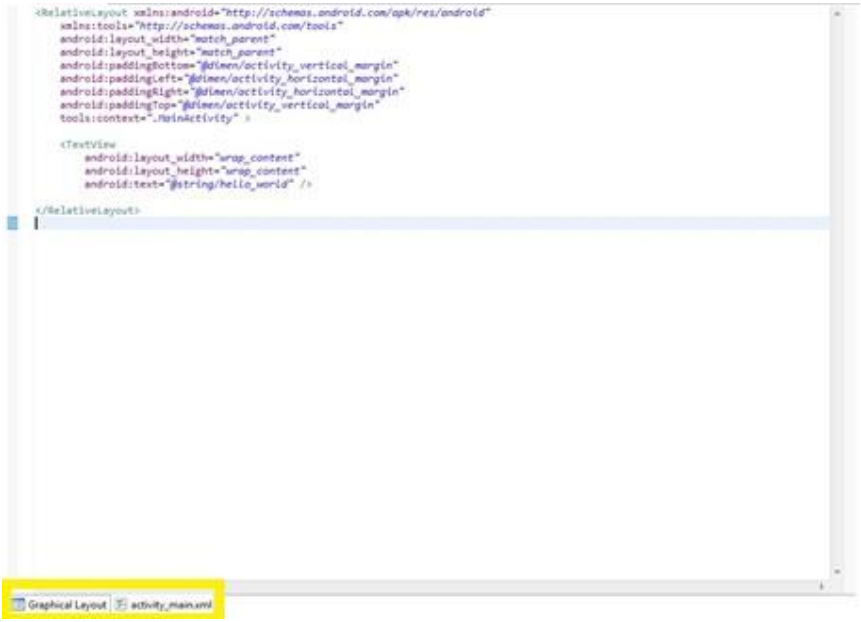


Figura 16. Programación escribiendo el código directamente.

En la parte izquierda de la programación mediante Graphical Layout nos aparece una columna bajo el nombre "Palette" (Figura 17) que incluye todo lo que podemos añadir a nuestra aplicación. Las opciones son múltiples, botones de diferentes tamaños, huecos para poder escribir, imágenes y más añadidos simples pero útiles.

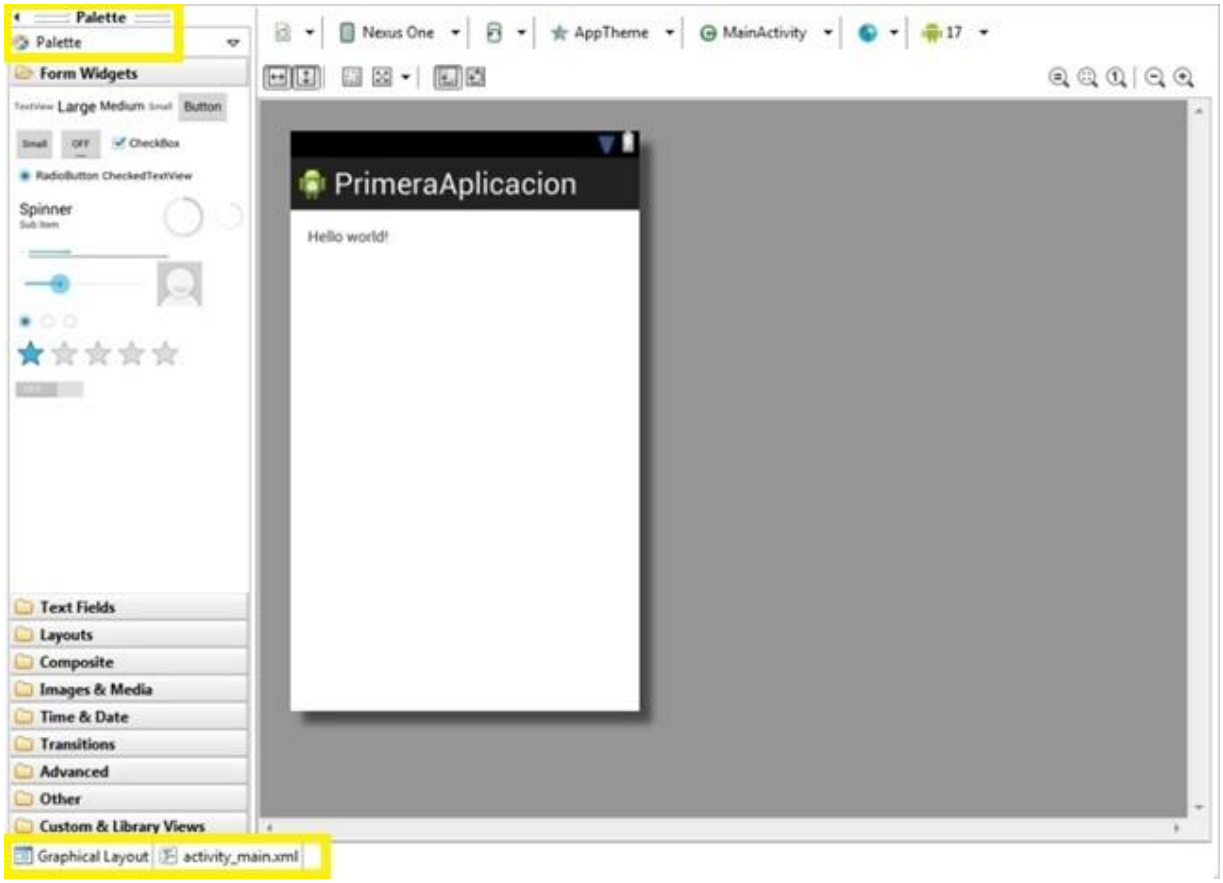


Figura 17. Programación a través de Graphical Layout.

Para añadir un botón, por ejemplo, debemos seleccionarlo y arrastrarlo hasta la pantalla (Figura 18). Automáticamente Eclipse generará el código en la pestaña "activity_main.xml" (Figura 19) y nos aparecerá el botón en pantalla. Pese a la comodidad en muchos casos que nos ofrece este método de programación, en ocasiones puede resultar problemático y siempre se recomienda programar todo mediante XML y utilizar el Graphical Layout sólo para verificar el aspecto final.

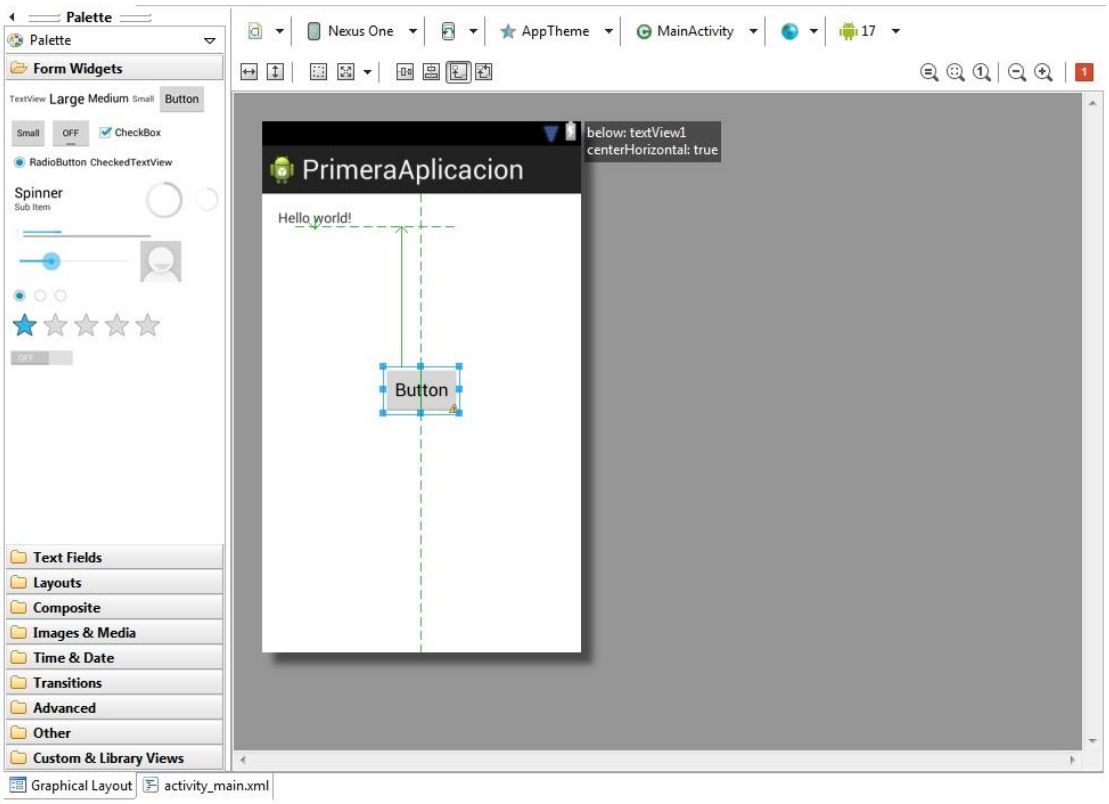


Figura 18. Botón añadido mediante Graphical Layout.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hello_world" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textView1"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="140dp"
        android:text="Button" />

</RelativeLayout>
```

Figura 19. Código generado automáticamente al incluir el botón.

2.6 CICLO DE VIDA DE UNA APLICACIÓN

La aplicación o actividad tiene cuatro estados: Activo, Pausado, Parado y Destruido. Explicaremos estas cuatro fases:

- **Activo:** La actividad se encuentra en primer plano, es la tarea más importante del dispositivo en ese momento y el usuario puede interactuar con ella.
- **Pausado:** Entramos en este estado cuando la actividad sigue siendo visible pero no totalmente por el usuario. Esto ocurre cuando otra aplicación se ejecuta y muestra, por ejemplo, un cuadro de dialogo que provoca la pérdida del foco por parte de nuestra aplicación pero no provoca su parada.
- **Parado:** La actividad no es visible para el usuario, queda a disposición del sistema el provocar su cierre definitivo para liberar memoria.
- **Destruida:** La actividad finalmente ha sido finalizada por parte del sistema Android y ya no se encuentra dentro de la pila de actividades.

La Figura 20 muestra un diagrama de flujo con el funcionamiento de una aplicación en condiciones normales:

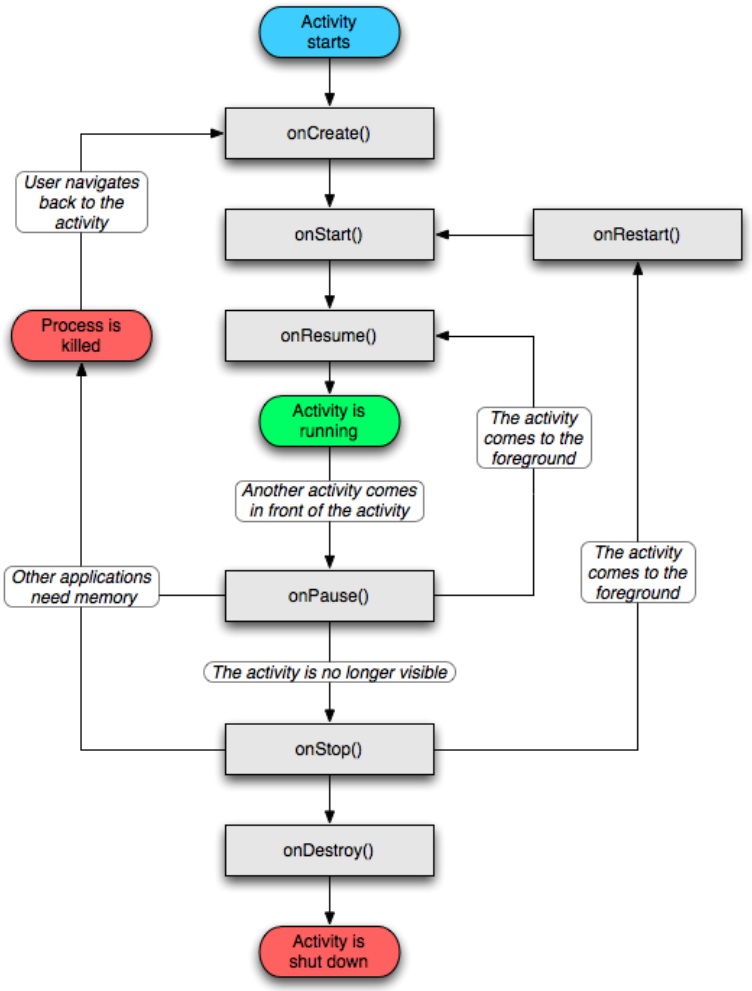


Figura 20. Ciclo de Vida de una aplicación.

Cada vez que una actividad cambia de estado se van a producir eventos que podrán ser capturados por ciertos métodos de la actividad. Los eventos hacen referencia a cualquier interacción que haga el sistema operativo Android sobre la aplicación, por ejemplo una llamada de teléfono, el bloqueo de la pantalla o que el usuario pulse el botón salir. Un método es cómo reacciona el programa a ese evento. En la Figura 20 aparecen los principales métodos de una aplicación y a continuación explicamos brevemente sus funciones [5]:

- **onCreate ( ):** Se utiliza para llamar a la creación de una actividad por primera vez. Después de esta llamada siempre se llama al onStart ( ).

- **onStart ( ):** Nos indica que la actividad está a punto de ser mostrada al usuario.

- **onResume ( ):** Se llama cuando la actividad va a comenzar a interactuar con el usuario.

- **onPause ( ):** Indica que la actividad actualmente en primer plano va a pasar a estar en segundo plano. Se debe parar cualquier animación o sonido en funcionamiento. Este método debe ejecutarse rápido ya que hasta que esto no ocurra no comenzará la nueva actividad.

- **onStop ( ):** Llamada cuando la actividad ya no es visible para el usuario porque otra actividad ha pasado a primer plano.

- **onRestart ( ):** Ocurre tras el onStop ( ), e indica que la actividad ha sido parada pero vuelve a estar en primer plano de nuevo.

- **onDestroy ( ):** También ocurre tras el onStop ( ) y es la llamada final de la actividad, después de ésta, es totalmente destruida. En el caso de que hubiese poca memoria es posible que la actividad se destruya completamente sin llamar a este método.

2.7 COMPONENTES DE UNA APLICACIÓN

Toda aplicación posee una serie de elementos clave para su desarrollo. Cada componente es un punto a través del cual el sistema puede entrar en la aplicación, aunque no todos son verdaderos puntos de entrada para el usuario. Existen cuatro componentes principales y cada uno tiene un ciclo de vida diferente que define cómo se crea y cómo se destruye:

Activities (Actividades): Representa una única pantalla de la aplicación, con su propia interfaz, con la que podremos interactuar. De todas las Actividades, una de ellas será considerada la principal y será la que se lance primero cuando ejecutemos la aplicación. Desde ésta podremos lanzar otras actividades secundarias que serán, el conjunto de ellas, las que conformen la aplicación.

Intents (Intenciones): Una intención hace referencia a la voluntad de realizar alguna acción, ya sea mandar un mensaje o introducir algún dato. Se encarga de notificar a las aplicaciones cambios en el estado del hardware, como es el introducir una tarjeta SD, avisar de datos entrantes, como la llegada de un mensaje o una llamada, y de eventos en aplicaciones, como cuando ejecutamos una. El sistema se encarga de elegir entre las actividades disponibles en el teléfono y dará respuesta a la intención con la actividad más adecuada.

Content Providers (Proveedores de contenido): Muchas aplicaciones que tenemos hoy en día en los teléfonos móviles comparten datos entre sí. Este elemento sirve para que se puedan compartir y se pueda acceder a la información. Los datos a los que se acceden pueden estar en el sistema, en una base de datos, en la web o en cualquier otro sistema de almacenamiento existente al que la aplicación en cuestión pueda acceder.

Services (servicios): Un servicio es un proceso que funciona de manera secundaria a la actividad principal, sin la necesidad de que el usuario interactúe con él. Un ejemplo claro de servicio es el reproductor de música que puede estar funcionando mientras nosotros realizamos otra actividad en el móvil.

3. DISEÑO DE LA APLICACIÓN

A lo largo de este capítulo desarrollaremos la idea de la aplicación por completo, explicando punto por punto los pasos que se han llevado a cabo hasta llegar al resultado final.

3.1 PLANTEAMIENTO

El objetivo del proyecto es realizar una aplicación que muestre unas imágenes de diseño mediante un carrusel que podremos ir desplazando a nuestra voluntad mediante la pantalla táctil. Además cada imagen tendrá asociada un texto que se mostrará cuando pulsemos en ella.

Las imágenes y textos que se han empleado para la realización de la aplicación forman parte de la obra creada por la artista Marina Anaya y la periodista Lidia Martín, que lleva por título 'Y si llueve' [6]. Se trata de un cuento de bocetos y poemas, un libro objeto de 20 fichas intercambiables. La aplicación pretende hacer llegar al usuario esta obra a través del dispositivo móvil, y permitirle visualizar cada una de estas fichas de forma interactiva.

Buscaremos la sencillez en el desarrollo de la aplicación, ya que un exceso de opciones o un menú demasiado denso alejan al usuario del objetivo primordial, que es la ejecución de la aplicación y la visualización de las imágenes.

El orden seguido para el desarrollo completo de la aplicación es el siguiente:

- Creación de una pantalla inicial de carga de la aplicación.
- Ejecutar el carrusel.
- Incorporar todas las imágenes con su correspondiente texto dentro de la aplicación.
- Aplicar el efecto de giro en tres dimensiones que nos permita rotar entre la vista de la imagen y la vista del texto.
- Añadir un botón para acceder a información extra sobre la aplicación.
- Añadir un botón que permita al usuario ir a la página web de la autora de los diseños.

3.2 IMÁGENES Y TEXTOS QUE COMPONEN EL CARRUSEL

El eje entorno al que gira todo el desarrollo y diseño de la aplicación son las imágenes y sus textos correspondientes. De la figura 21 a la 40 se muestran todas ellas y su texto al lado.

Figura 21. No atiendo a razones, sólo a ti.

Figura 22. De haberlo sabido habría hecho lo mismo.

Figura 23. Estoy colada por ti, diluida de los pies a la cabeza.

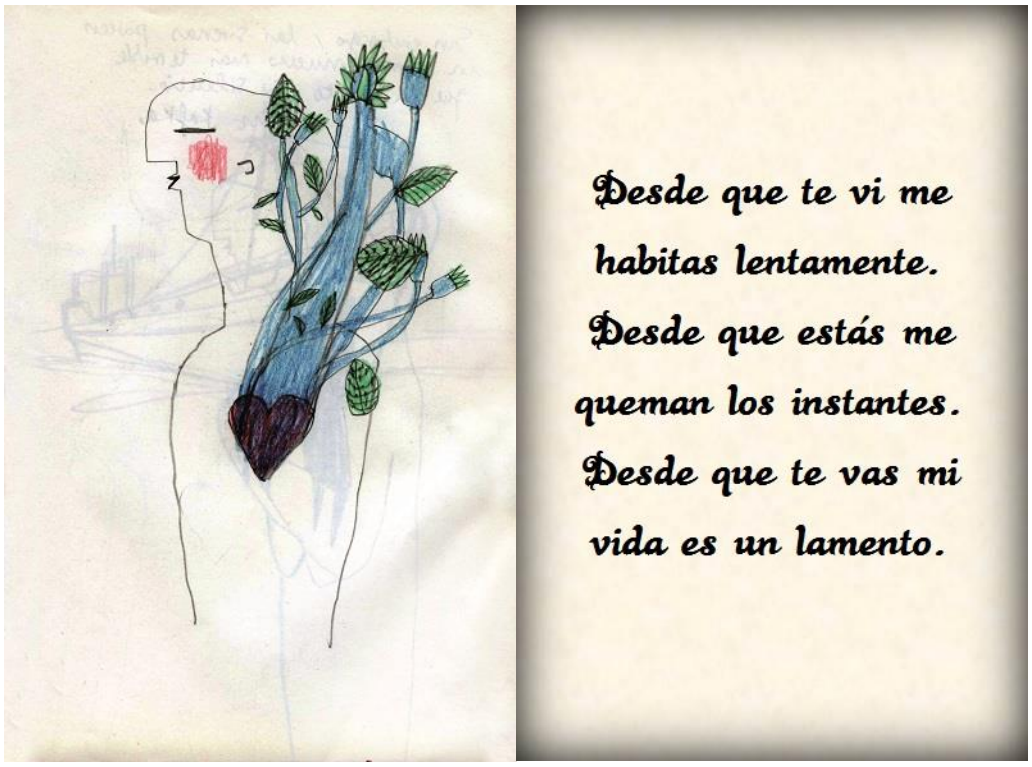


Figura 24. Desde que te vi me habitas lentamente. Desde que estás me queman los instantes. Desde que te vas mi vida es un lamento.

Figura 25. Si vuelves a huir amenazo con volverte a rechazar.

Figura 26. Y con el tiempo fue amargando tu sabor.

Figura 27. Llevabas el pelo suelto tapando tu cara y un jersey hasta el cuello de dudas.

Figura 28. Riégame. No dejes que me seque. No querrás que me remate el asfalto.

Figura 29. Aullábamos como lunáticos en noches de locura llena.

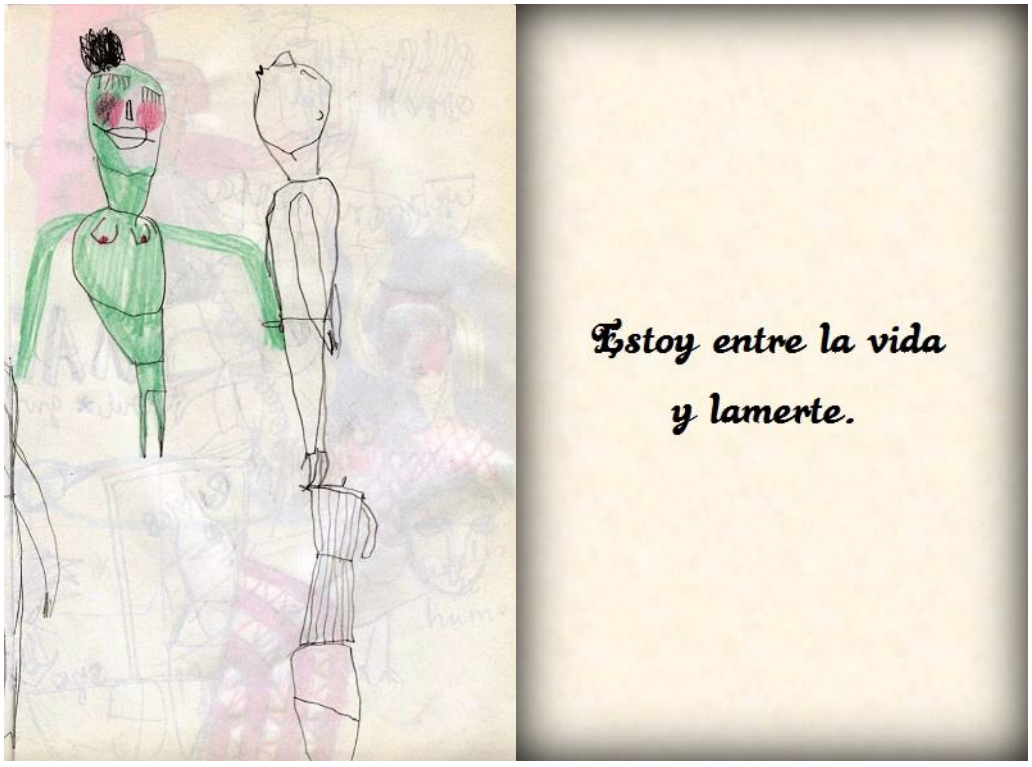


Figura 30. Estoy entre la vida y lamerte.

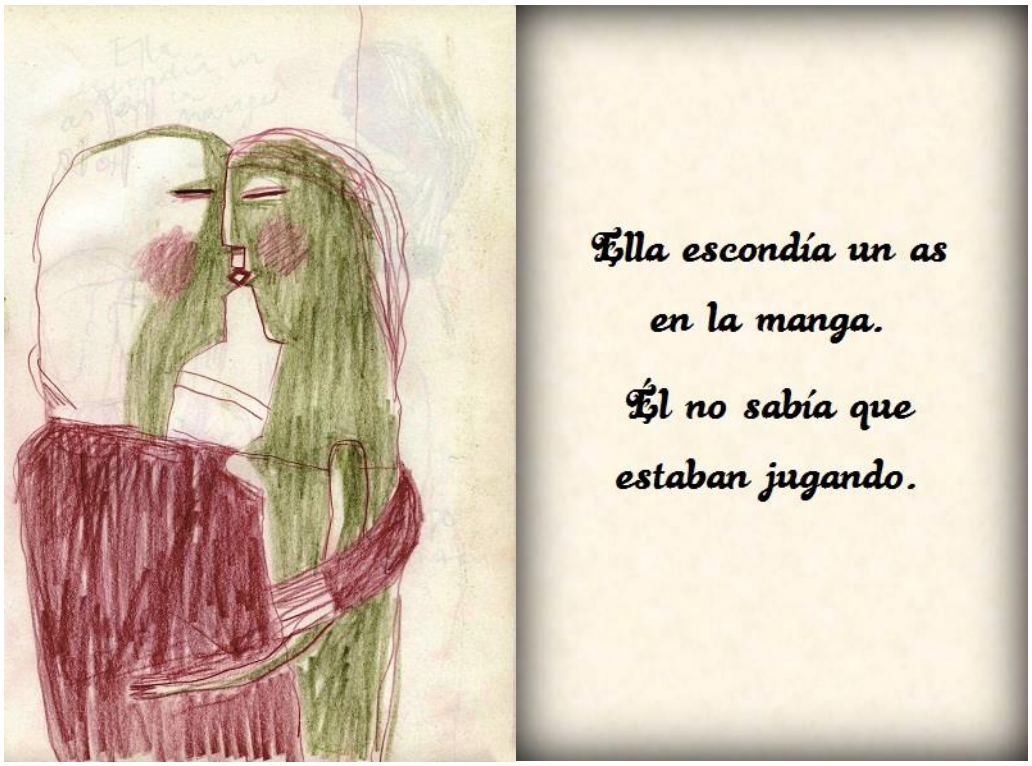


Figura 31. Ella escondía un as en la manga. Él no sabía que estaban jugando.

Figura 32. Me vendiste humo y te ahogaste de tanto mentir.

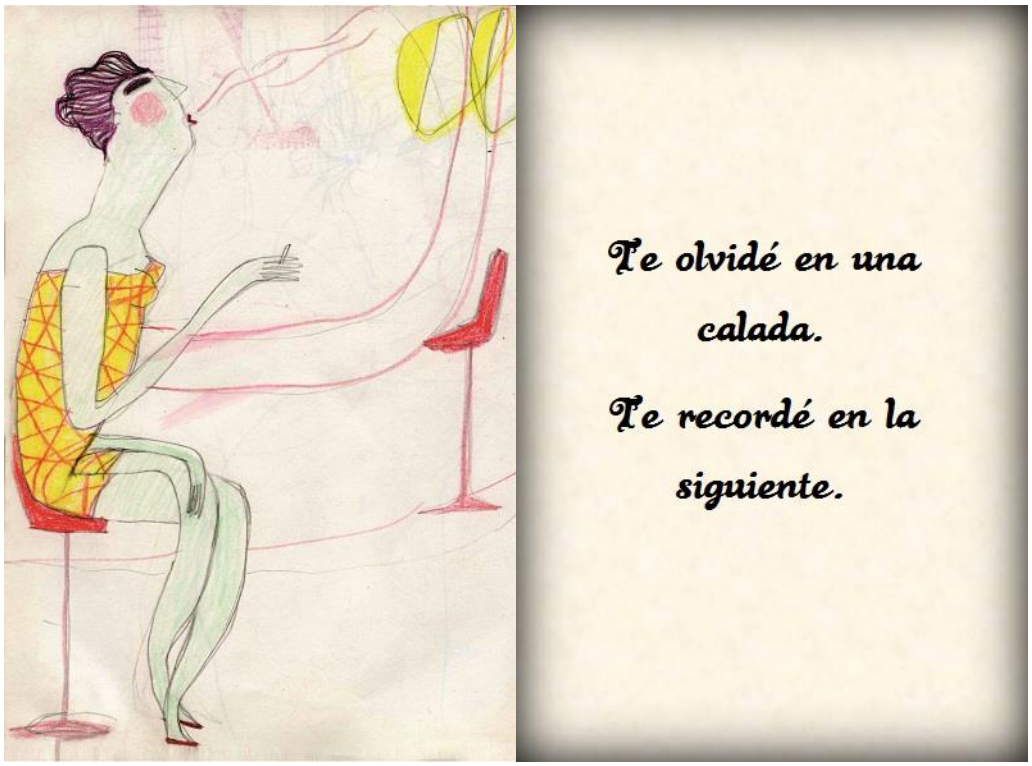


Figura 33. Te olvidé en una calada. Te recordé en la siguiente.

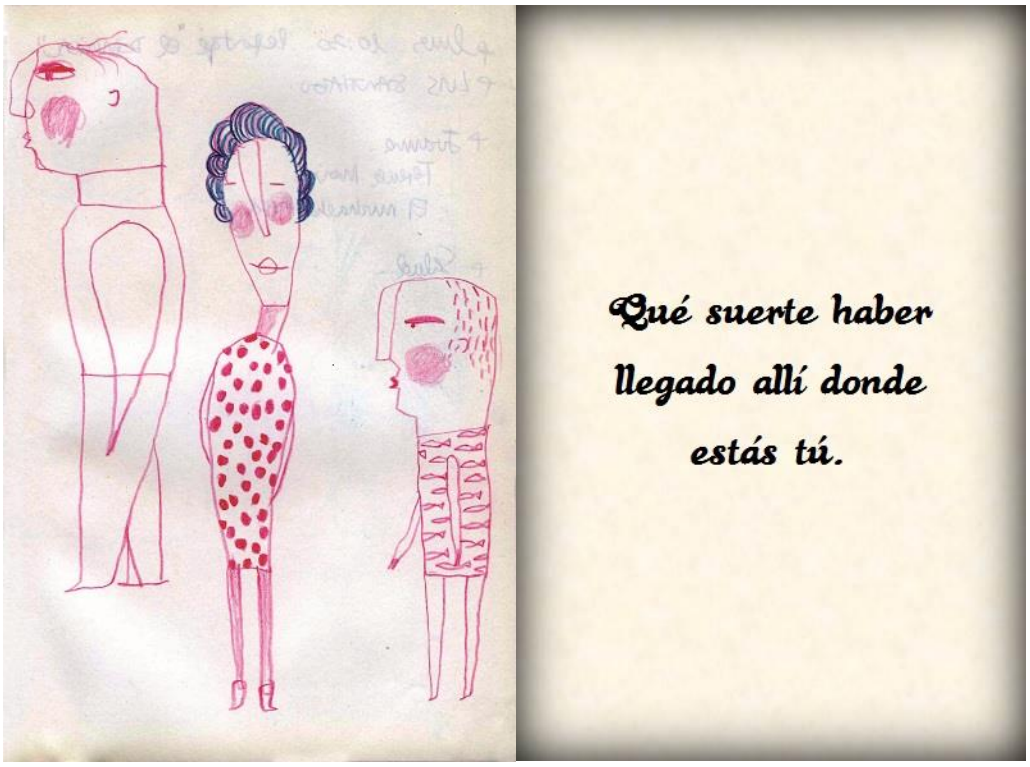


Figura 34. Qué suerte haber llegado allí donde estás tú.

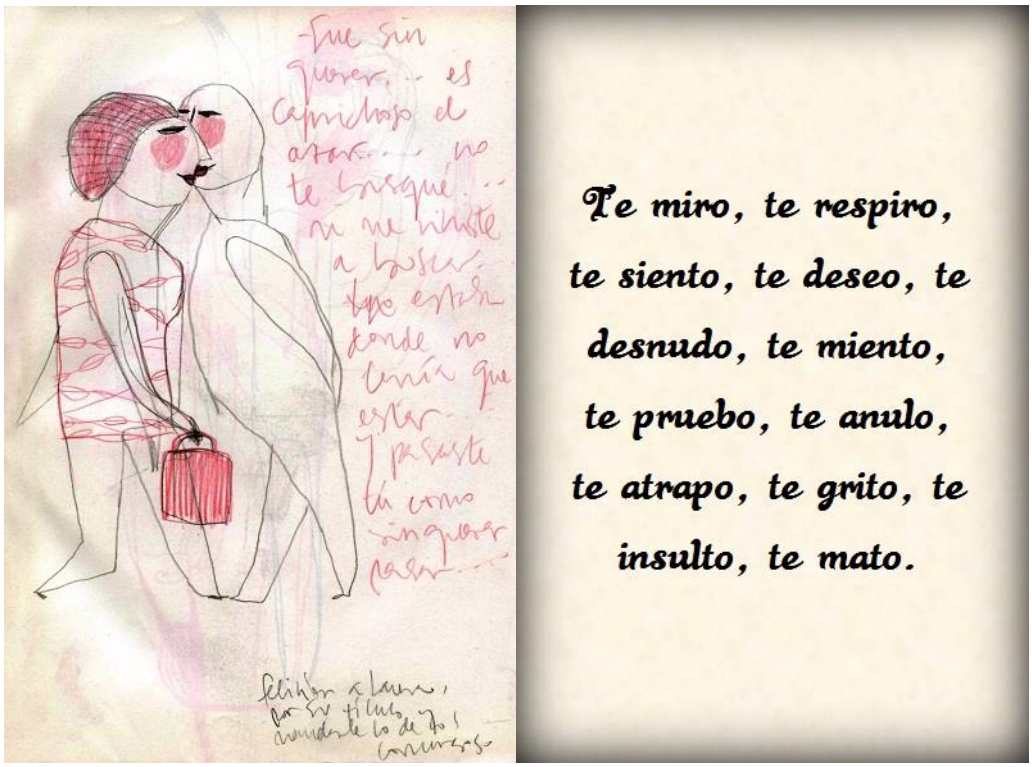


Figura 35. Te miro, te respiro, te siento, te deseo, te desnudo, te miento, te pruebo, te anulo, te atrapo, te grito, te insulto, te mato.

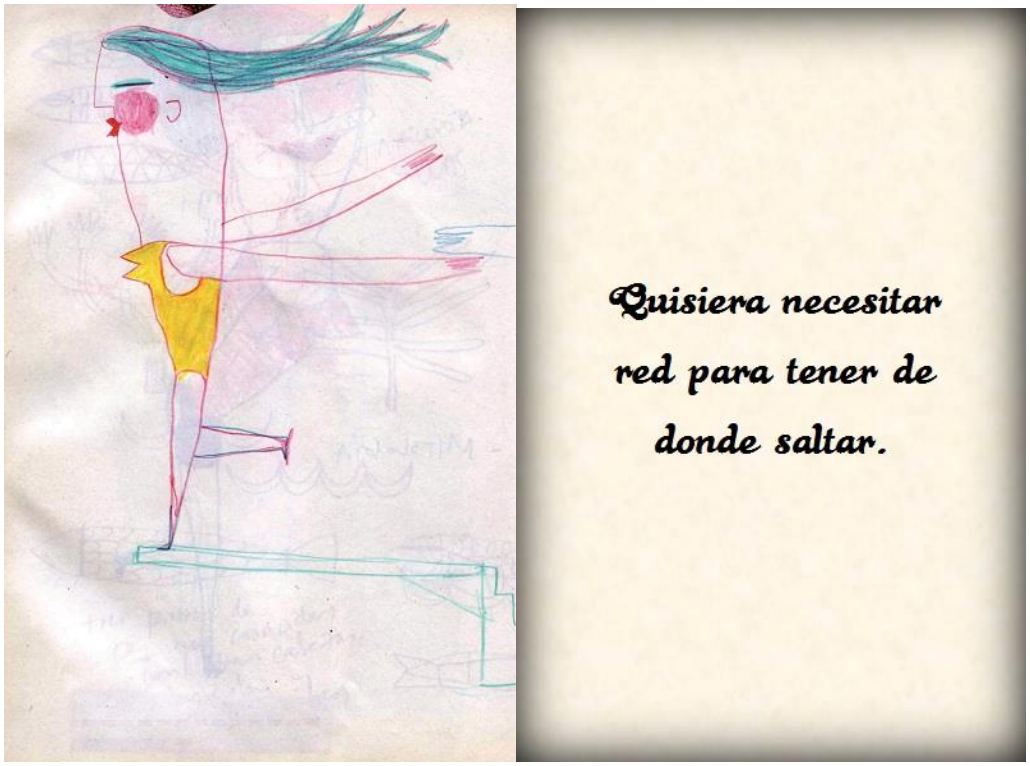


Figura 36. Quisiera necesitar red para tener de donde saltar.

Figura 37. Tócame otra vez, Sam.

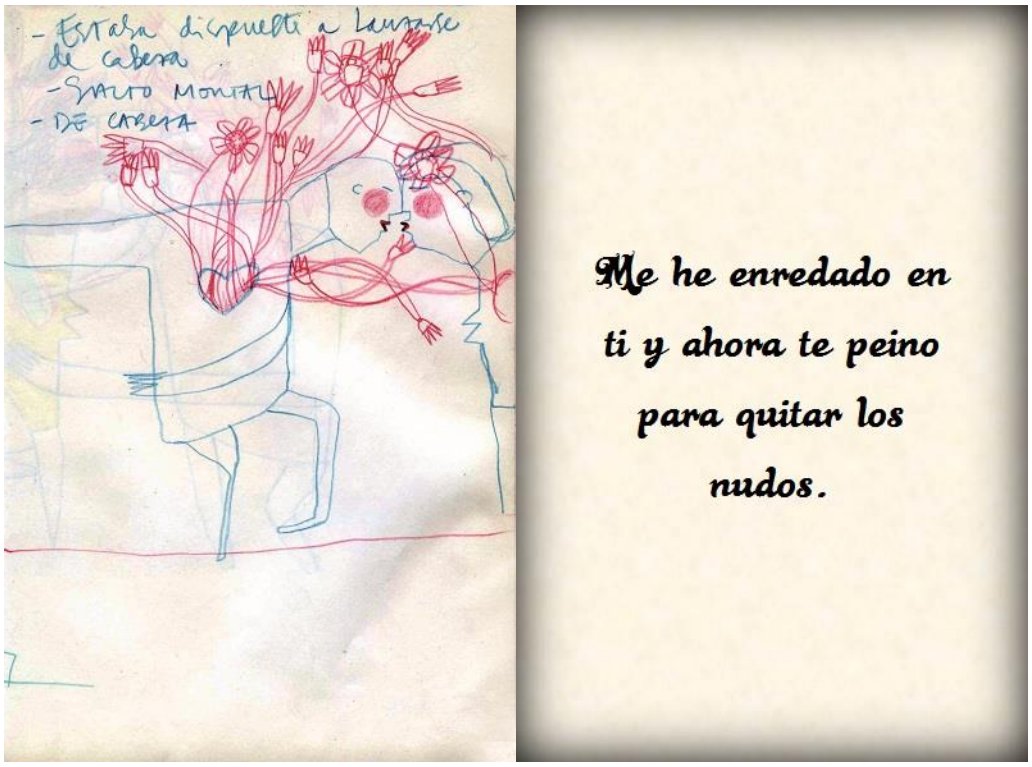


Figura 38. Me he enredado en ti y ahora te peino para quitar los nudos.

Figura 39. Quiero volver a casa por la curva de tu espalda.

Figura 40. Y si llueve siento que no soy el único que llora.

3.3 CONCEPTOS BÁSICOS DE LA PROGRAMACIÓN ANDROID

Es importante diferenciar entre los dos componentes principales de una aplicación en Android: los archivos xml y los archivos java. Los archivos Java se encuentran en la carpeta "src" y alojan el código fuente de nuestra aplicación y por lo tanto son los encargados de indicar a nuestra aplicación cual es el correcto funcionamiento de nuestra aplicación. Los archivos con extensión xml se encuentran dentro de la subcarpeta "Layout" de la carpeta "res" y definen la interfaz gráfica de las actividades de la aplicación.

Además de estos dos componentes, existe un tercero que es el "AndroidManifest.xml". Es un archivo en el que se declaran las especificaciones de nuestra aplicación. Esto comprende información como las versiones de Android para las que es compatible, el nombre e icono de la aplicación, la actividad principal que se ejecuta o como se muestra la aplicación, es decir, si es a pantalla completa, horizontal, vertical, etc.

En los dos párrafos anteriores se menciona el concepto actividad. Una actividad es la encargada de mostrar el interfaz del usuario e interactuar con él. Para ejecutar una actividad se utiliza el método "onCreate" acompañado de "setContentView", acompañado de la referencia del archivo xml que quiera ejecutar el desarrollador [7].

Para entender estos conceptos se analizará una aplicación simple. Estará compuesta por un botón que abre una nueva actividad y ésta mostrará un texto. En la Figura 41 se muestra el "MainActivity.java". Resaltado aparecen los dos métodos que lo componen: en negro el "onCreate" encargado de ejecutar el archivo "activity_main.xml" y en amarillo el que provoca que se ejecute la segunda actividad. Cuando el usuario pulse el botón que aparece en el activity_main.xml automáticamente se abrirá el archivo "Texto.Java".

```
package com.example.helloworld;

import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.View;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void abrir (View view) {

        Intent i = new Intent (this, Texto.class) ;
        startActivity (i);
    }
}
```

Figura 41. Código presente en el MainActivity.java.

En la Figura 42 se muestra el "activity_main.xml" que, como hemos comentado antes, se ejecuta al iniciar la aplicación por medio del método onCreate presente en el MainActivity.Java. El recuadro negro resalta la información sobre el diseño del botón, su posición respecto a la pantalla, nombre y dimensiones. Además se incluye el evento "onClick" es decir, que método se ejecutará cuando el usuario pulse el botón. El nombre que recibe hace referencia al método mostrado en la Figura 40 y que genera que se abra la nueva actividad.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="145dp"
        android:onClick="abrir"
        android:text="Button" />

</RelativeLayout>
```

Figura 42. Código presente en el archivo activity_main.xml.

Estos son los conceptos básicos de programación en Android. A medida que se avanza en el diseño de la aplicación la variedad de métodos y la dificultad de las actividades se ve aumentada.

3.4 PANTALLA INICIAL

Figura 43. Pantalla inicial de la aplicación.

Cuando se ejecuta la aplicación se decidió que se mostrará una imagen con información básica sobre ella. La imagen corresponde a uno de los veinte diseños del carrusel y la información que se incluye son tanto el nombre de la aplicación como el nombre de las autoras (Figura 43). La carga de esta imagen se realiza durante tres segundos y posteriormente se muestra ya el carrusel.

En la Figura 44 se muestra el archivo java de la actividad inicial de nuestra aplicación y que ejecuta la pantalla inicial. En el primer recuadro se muestra el tiempo que queremos que dure esta imagen. El segundo recuadro muestra el método que abre la actividad del Carrusel.java tras cumplirse el tiempo indicado.

```
package com.proyecto.carruselpfg;

import android.app.Activity;

public class MainActivity extends Activity {

    private static int SPLASH_TIME_OUT = 3000;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        this.requestWindowFeature(Window.FEATURE_NO_TITLE);
        setContentView(R.layout.activitysplash);

        new Handler().postDelayed(new Runnable() {


            @Override
            public void run() {
                Intent i = new Intent(MainActivity.this, Carrusel.class);
                startActivity(i);

                finish();
            }
        }, SPLASH_TIME_OUT);
    }

}
```

Figura 44. Código correspondiente al ActivityMain.java.

La Figura 45 corresponde al activitysplash.xml que se ejecuta durante la duración estimada. La información que aparece es relativa a la imagen escogida.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/andr
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <ImageView
        android:contentDescription="@string/desc"
        android:id="@+id/imageView1"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:background="@color/black"
        android:src="@drawable/splash" />

</RelativeLayout>
```

Figura 45. Código correspondiente al activitysplash.xml.

3.5 CARRUSEL DE IMÁGENES

Figura 46. Vista en la aplicación del carrusel de imágenes.

El carrusel de imágenes (Figura 46) es la parte fundamental de nuestro proyecto y también la parte más compleja. Su diseño se puede dividir en dos partes: el carrusel con las veinte imágenes y el efecto de giro en tres dimensiones que nos permite ver el texto asociado a cada imagen.

```
import android.app.Activity;

public class Carrusel extends Activity {

    ImageView lastClicked = null;
    int padding = 5;
    Animation rotacion;
    Context c;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        c = this;
        setContentView(R.layout.iniciar);
        LinearLayout l;
        l = (LinearLayout) findViewById(R.id.carrusel);
        final int[] images = new int[] { R.drawable.imagen1, R.drawable.imagen2, R.drawable.imagen3,
                R.drawable.imagen4, R.drawable.imagen5,R.drawable.imagen6,R.drawable.imagen7,
                R.drawable.imagen8, R.drawable.imagen9,R.drawable.imagen10,R.drawable.imagen11,
                R.drawable.imagen12, R.drawable.imagen13,R.drawable.imagen14,R.drawable.imagen15,
                R.drawable.imagen16, R.drawable.imagen17,R.drawable.imagen18,R.drawable.imagen19,
                R.drawable.imagen20 };
        final int[] images_back = new int[] { R.drawable.texto1, R.drawable.texto2, R.drawable.texto3,
                R.drawable.texto4, R.drawable.texto5,R.drawable.texto6,R.drawable.texto7,
                R.drawable.texto8, R.drawable.texto9,R.drawable.texto10,R.drawable.texto11,
                R.drawable.texto12, R.drawable.texto13,R.drawable.texto14,R.drawable.texto15,
                R.drawable.texto16, R.drawable.texto17,R.drawable.texto18,R.drawable.texto19,
                R.drawable.texto20 };

        final String[] cards_descriptions = getResources().getStringArray(R.array.card_description_array);

        for (int i = 0; i < cards_descriptions.length; i++) {

            LayoutInflater inflater = (LayoutInflater)getBaseContext().getSystemService(Context.LAYOUT_INFLATER_SERVICE);
            View v = inflater.inflate(R.layout.card, null);
            final ImageView dibujo_delante = (ImageView) v.findViewById(R.id.CardImage);
            dibujo_delante.setImageResource(images[i]);
            dibujo_delante.setPadding(padding, 25, padding, 0);
            final ImageView dibujo_detras = (ImageView) v.findViewById(R.id.CardImageBack);
            dibujo_detras.setImageResource(images_back[i]);
            dibujo_detras.setPadding(padding, 25, padding, 0);
            v.setOnClickListener( new OnClickListener() {
```

Figura 47. Código correspondiente al Carrusel.

En la Figura 47 se muestra el código correspondiente al carrusel. La carga de imágenes y textos se ha realizado mediante un array que va recorriendo toda la información para luego mostrarla de manera ordenada y consecutiva. Esto nos permite asociar de manera sencilla las imágenes con sus textos correspondientes. Todo se almacena en un Linearlayout (señalizado en el recuadro de la Figura 47), que no es más que un contenedor de información que en este caso guarda los datos de manera lineal, es decir, uno a continuación del otro [8]. Esta información es pasada al archivo iniciar.xml que contendrá la interfaz gráfica del carrusel (Figura 48).

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="bottom"
    android:background="@color/black" >

    <HorizontalScrollView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_below="@id/imageView1"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:layout_margin="0dp"
        android:scrollbars="none" >

        <LinearLayout
            android:id="@+id/carrusel"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="horizontal" >
        </LinearLayout>
    </HorizontalScrollView>
```

Figura 48. Código en el archivo iniciar.xml.

A la hora de plantear como guardar los textos se optó por crear una imágenes de igual tamaño que los diseños y almacenarlos superpuestos, de tal manera que sólo se viese uno. Realmente es como si estuviesen almacenados dos carruseles, uno encima del otro. Esto permite introducir el efecto del giro en tres dimensiones que se explica en el siguiente punto.

3.6 EFECTO GIRO DE TRES DIMENSIONES A LAS IMÁGENES

Para realizar el efecto de giro en tres dimensiones se ha usado una aplicación simple ya existente que realizaba únicamente este efecto para una sola imagen [9]. Para poder aplicar este efecto a todas las imágenes primer fue necesario realizar un sistema por el cual nuestra aplicación supiese diferenciar el estado en el que se encontraba la imagen, es decir, si estaba mostrando el diseño o el texto (Figura 49).

```
        public void onClick(View v) {

            TextView card_state = (TextView) v.findViewById(R.id.card_state);
            boolean anverse = false;

            if (card_state.getText().toString().equals(("0")))
            {
                dibujo_detras.setVisibility(View.INVISIBLE);
                anverse = false;
                card_state.setText("1");
                applyRotation(0, 270, anverse, dibujo_detras, dibujo_delante);

            }
            else
            {
                dibujo_delante.setVisibility(View.INVISIBLE);

                anverse = true;
                card_state.setText("0");
                applyRotation(0, -270, anverse, dibujo_detras, dibujo_delante);

            }

        }
    });
```

Figura 49. Código del archivo Carrusel.java encargado de posicionar las imágenes.

Además en esta parte también se incluye las propiedades que vamos a dar al giro de la imagen. Por ejemplo en nuestro caso aplicamos un giro de 270º en sentido horario cuando se pasa de mostrar la imagen al texto y un giro de 270º en sentido antihorario para el caso contrario. Además en el recuadro vemos la inclusión de la función "applyRotation" que es la encargada de enlazar el efecto de giro con nuestra aplicación, como se muestra en la Figura 50.

```
private void applyRotation(float start, float end, boolean isFirstImage, ImageView image1, ImageView image2) {

    final float centerX = image1.getWidth() / 2.0f;
    final float centerY = image1.getHeight() / 2.0f;


    final Flip3dAnimation rotation =
    new Flip3dAnimation(start, end, centerX, centerY);
    rotation.setDuration(1000);
    rotation.setFillAfter(true);
    rotation.setInterpolator(new AccelerateInterpolator());
    rotation.setAnimationListener(new DisplayNextView(isFirstImage, image1, image2));

    if (isFirstImage)
    {
        image1.startAnimation(rotation);
    } else {
        image2.startAnimation(rotation);
    }

}
```

Figura 50. Código del archivo Carrusel.java encargado de la rotación de las imágenes.

Con esto conseguimos el efecto deseado de giro en nuestras imágenes de tres dimensiones y además que la aplicación memorice el estado de cada una de las cartas y actúe en consecuencia (Figura 51 y Figura 52).

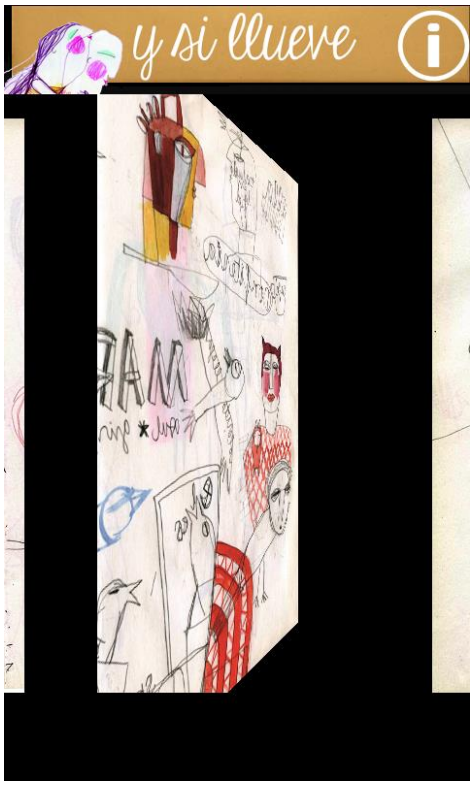


Figura 51. Muestra 1 del giro en tres dimensiones.

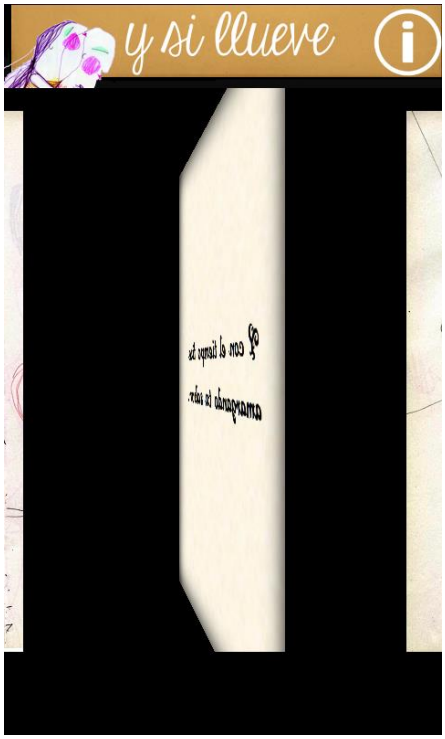


Figura 52. Muestra 2 del giro en tres dimensiones.

3.7 INCLUSIÓN DE UN BOTÓN "SOBRE LA APLICACIÓN"

Para dar información extra al usuario sobre la aplicación, se ha incluido un botón en la parte superior de la ventana del carrusel, que permite acceder a una nueva actividad con datos sobre la aplicación (Figura 53).

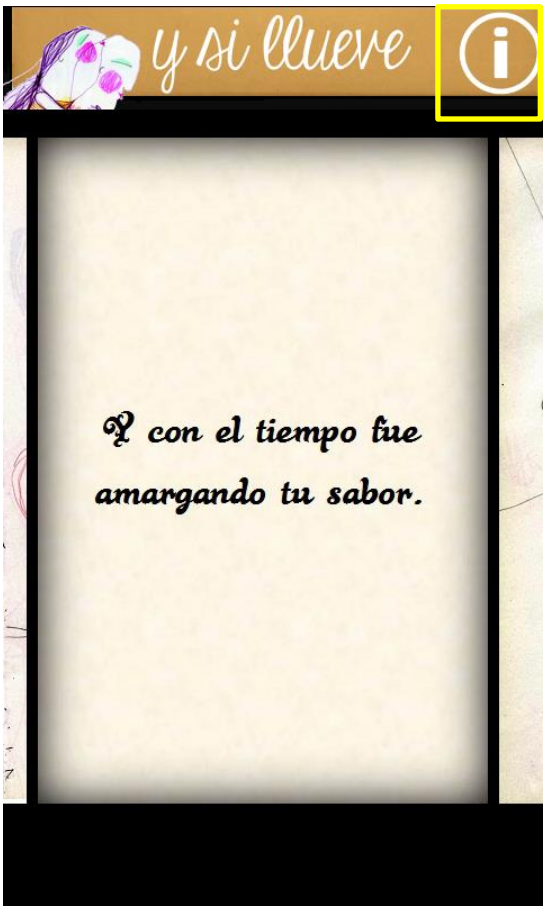


Figura 53. Imagen del botón "Sobre la aplicación".

La programación del funcionamiento de este botón se realiza desde el archivo Carrusel.java. Es ahí donde programamos para que nos lance a una nueva actividad en la que se muestre la ventana con la información.

```
public void sobreabrir (View view) {

        Intent i = new Intent (this, SobreLaAplicacion.class) ;
        startActivity (i);
    }
```

Figura 54. Lanzamiento de la actividad "SobreLaAplicacion".

El diseño de la nueva actividad está compuesto por tres partes: una imagen en la parte de arriba, un texto con información sobre cómo se crearon los diseños y los textos y por último un enlace directo a la página web de la autora (Figura 55).

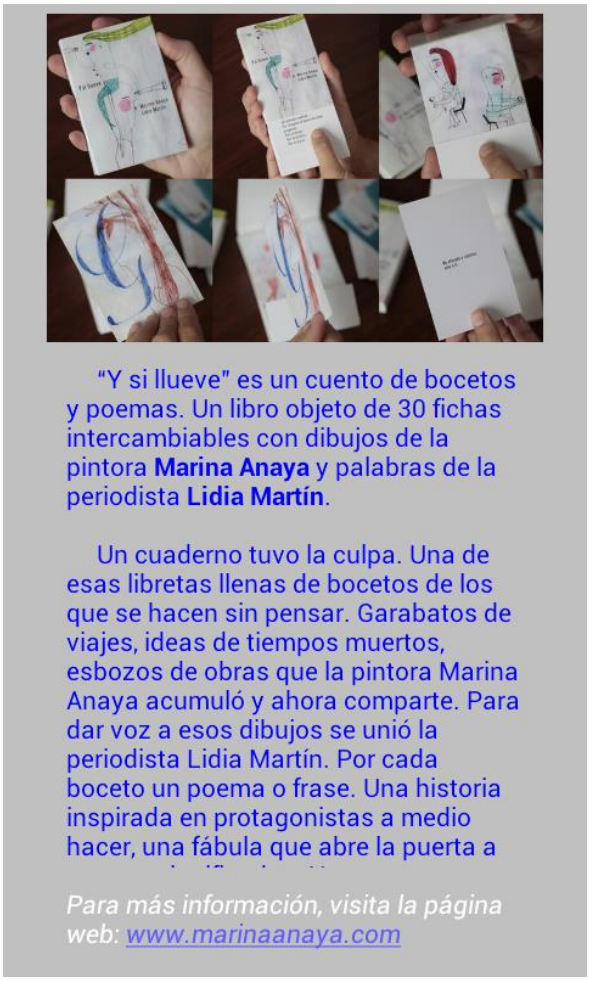


Figura 55. Ventana "Sobre la aplicación".

En la Figura 56 se muestra las tres partes de las que está compuesta esta actividad. En el primer recuadro se muestra el diseño de la imagen superior. En el segundo recuadro aparece el texto informativo. En el se incluye el comando "ScrollView" que se utiliza para que el usuario pueda deslizar el texto de manera vertical y poder verlo todo. Por último el tercer recuadro contiene la información correspondiente al enlace directo a la página web de la creadora de los diseños.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom"
    android:background="@color/silver" >

    <ImageView
        android:contentDescription="@string/desc"
        android:id="@+id/imageView1"
        android:layout_width="90dp"
        android:layout_height="180dp"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:layout_marginTop="5dp"
        android:src="@drawable/sobrefoto" />

    <ScrollView
        android:layout_width="fill_parent"
        android:layout_height="200dp"
        android:layout_above="@+id/paginaweb"
        android:layout_below="@+id/imageView1"
        android:padding="10dip" >

        <TextView
            android:id="@+id/about_content"
            android:layout_width="250dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:text="@string/textosobre"
            android:textColor="@color/blue" />

    </ScrollView>

    <TextView
        android:id="@+id/paginaweb"
        android:layout_width="250dp"
        android:layout_height="50dp"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:text="@string/web"
        android:textColor="@color/white"
        android:textSize="14sp"
        android:textStyle="italic"
        android:typeface="normal"
        android:autoLink = "web" />

</RelativeLayout>
```

Figura 56. Código del archivo "sobrelaaplicacion.xml".

3.8 ICONO DE LA APLICACIÓN

Para realizar el icono de la aplicación se ha escogido un fragmento de una de las imágenes, ya que así será fácilmente reconocible por el usuario asociar la imagen con la aplicación (Figura 57). Se le ha añadido un marco para hacerla más agradable estéticamente.

Figura 57. Icono final de la aplicación.

Al crear el proyecto al comienzo, Eclipse te genera un icono por defecto que es el logotipo de Android. Este se puede cambiar por parte del usuario en el momento en que se desee. Para ello, debemos adjuntar nuestra imagen dentro de una de las carpetas "drawable" y posteriormente debemos ir al archivo "AndroidManifest.xml" (Figura 58).

Dentro de este archivo encontramos diferentes líneas de programación y debemos buscar aquella que hace referencia al icono: "Android:icon". Para cambiarla simplemente debemos poner a continuación "@drawable/" y el nombre de la imagen que hayamos seleccionado, en nuestro caso "portada".

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.proyecto.carruselpfg"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/portada"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" android:debuggable="true">
        <activity
            android:name="com.proyecto.carruselpfg.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name="SobreLaAplicacion"></activity>
                <activity android:name="Carrusel"></activity>
```

Figura 58. Resaltada la parte que se ha cambiado del AndroidManifest.xml.

Una vez hecho esto y guardando el proyecto, podemos ejecutar el emulador para verificar que en nuestro teléfono virtual realmente aparece ya nuestro nuevo icono (Figura 59).

Figura 59. Imagen del menú del teléfono virtual.

3.9 TRADUCCIÓN DE LA APLICACIÓN

Con el objetivo de llegar al mayor número de usuarios posibles, se ha decidido ofrecer la aplicación en dos idiomas distintos, inglés y español. Cualquier aplicación lleva por defecto un idioma que será el que el desarrollador quiera. Después se pueden añadir otros, para que se ejecuten en el caso de que la aplicación sea descargada por un usuario cuyo dispositivo use otro idioma. Es decir, si la aplicación tiene el inglés como idioma por defecto y además el español, en el caso de que se descargue en un teléfono cuyo idioma por defecto sea este último, automáticamente se cambiará. Si el idioma del dispositivo no fuese ninguno de los que se incluyen en la aplicación, se mantendría el idioma predeterminado por la aplicación, en nuestro caso el inglés.

Para poder traducir la aplicación debemos crear una carpeta extra dentro de la carpeta "res" (Figura 60). El nombre de esta carpeta será "values" un guion y continuación el código del lenguaje al que vamos a realizar la traducción. Estos códigos corresponden a los establecidos por la ISO 639-1. Deben ponerse de manera correcta, ya que si no fuese así, la aplicación no lo reconocería. Buscando el código correspondiente al español quedaría "values-es".

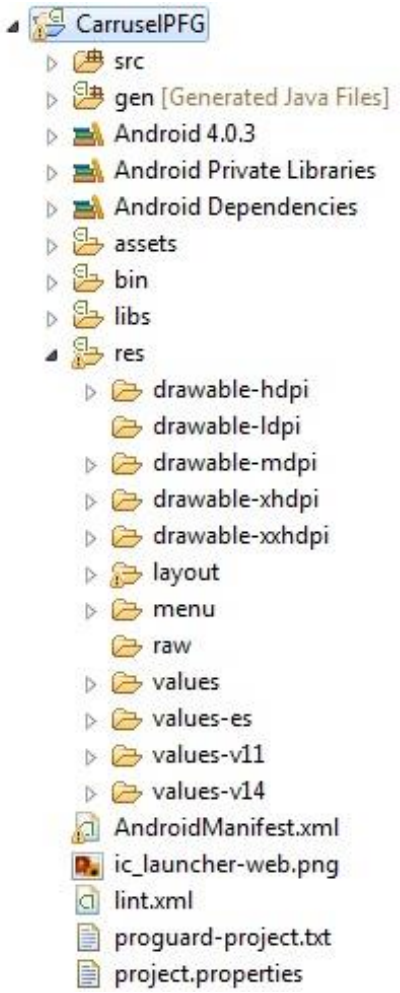


Figura 60. Proyecto con la carpeta values-es.

Dentro de la nueva carpeta debemos crear un archivo xml con el nombre "strings", ya que es el que se usa para hacer referencia a las cadenas de caracteres que hay en nuestra aplicación. Para facilitar la traducción y evitar que se nos olvide algún apartado por traducir, lo recomendable es copiar el archivo strings.xml de la carpeta "values" original (la que tiene el idioma por defecto) a la nueva carpeta, para después ir traduciendo cada uno de los textos.

Finalmente, una vez realizada la traducción tendríamos ya la aplicación disponible en dos idiomas distintos, tanto en español (Figura 61) como en inglés (Figura 62).

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">Y si llueve</string>
    <string name="menu_settings">Opciones</string>
    <string name="iniciar">Iniciar</string>
    <string name="textosobre"> \t "Y si llueve" es un cuento de bocetos y poemas. Un
    <string name="web">Para más información, visita la página web: www.marinaanaya.co
    <string name="volver">Volver</string>
    <string name="desc">I</string>
    <string name="volveralmenu">Volver al menú</string>
    <string name="saliraplicacion">Salir</string>

</resources>
```

Figura 61. Textos traducidos al español.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">And if it rains</string>
    <string name="menu_settings">Settings</string>
    <string name="iniciar">Start</string>
    <string name="sobrelaaplicacion">About the application</string>
    <string name="textosobre"> \t "And if it rains" is a tale of drafts and poems.
    <string name="web">For more information, visit the website: www.marinaanaya.cc
    <string name="volver">Back</string>
    <string name="desc">I</string>
```

Figura 62. Textos traducidos al inglés.

Este método tan sencillo para cambiar el idioma permite al desarrollador ir incluyendo nuevos idiomas si le resulta interesante para poder llegar a un mercado más amplio con su aplicación.

3.10 CAMBIAR EL NOMBRE A LA APLICACIÓN

Cuando creamos un nuevo proyecto en Android, automáticamente Eclipse designa el mismo nombre para la aplicación que estemos desarrollando. Sin embargo puede que con el avance del proyecto se quiera cambiar este por otro más adecuado. Esto ocurre en nuestro caso. En un comienzo el nombre del proyecto incluye las siglas PFG (Proyecto Fin de Grado) que no resulta adecuado al subir la aplicación a Google Play.

Para cambiar el nombre de la aplicación debemos saber que este viene definido en el "AndroidManifest.xml" bajo el comando "android:label = "@string/ app_name". Por lo tanto, para cambiar el nombre sólo debemos de ir al archivo "string.xml" y cambiar la cadena de caracteres que corresponde a "app_name".

El nombre final que recibirá la aplicación es "Y si llueve", que es el nombre que recibe la colección de todas las cartas. Su versión en inglés se traduciría como "And if it rains". En la Figura 63 se muestra el cambio ya realizado.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">Y si llueve</string>
    <string name="menu_settings">Opciones</string>
    <string name="iniciar">Iniciar</string>
    <string name="textosobre"> \t "Y si llueve" es un cue
    <string name="web">Para más información, visita la pá
    <string name="volver">Volver</string>
    <string name="desc">I</string>
    <string name="volveralmenu">Volver al menú</string>
    <string name="saliraplicacion">Salir</string>

</resources>
```

Figura 63. Cambio del nombre de la aplicación.

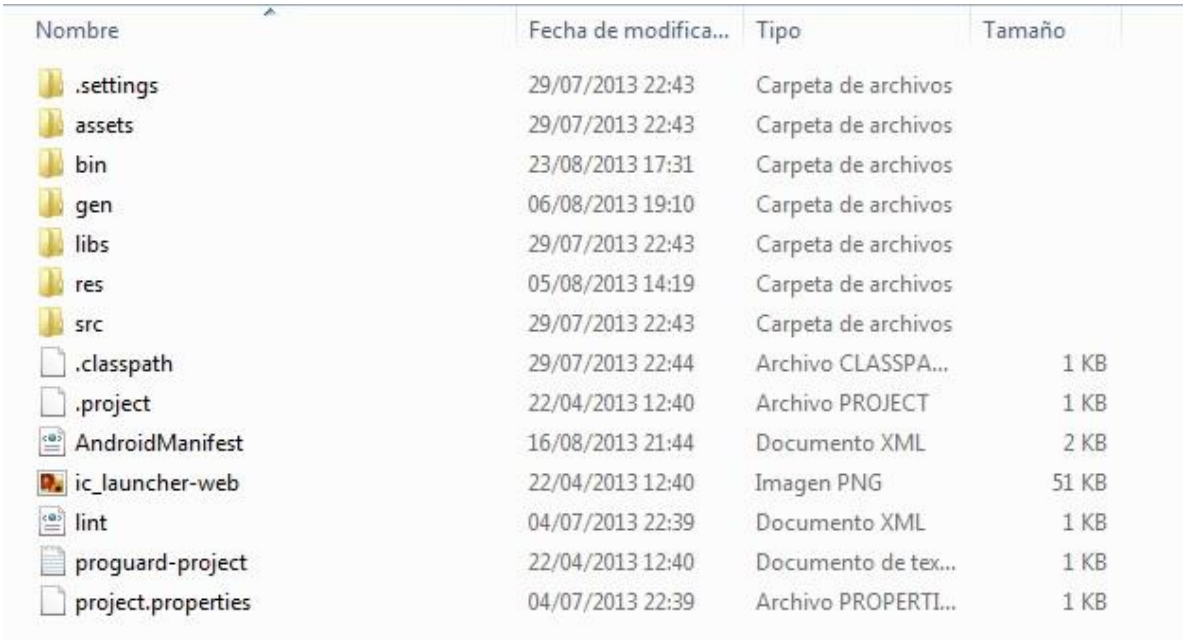
Una vez hecho esto, una vez ejecutemos la aplicación en nuestro móvil virtual ya aparecerá el cambio realizado para todas las pantallas de nuestra aplicación y para el texto que aparece debajo del icono (Figura 64).

Figura 64. Texto final de la aplicación.

3.11 PRUEBA EN UN DISPOSITIVO MÓVIL

Para verificar el correcto funcionamiento de la aplicación, a lo largo del desarrollo se ha usado el Android Virtual Device, por comodidad y velocidad. Sin embargo, es aconsejable a lo largo del proceso ir probando de vez en cuando la aplicación en el móvil, ya que va a ser el medio final en el que va a estar presente. En este caso las pruebas se han ido haciendo en un Samsung Galaxy S II i9100. Posee una pantalla táctil de 4,3 pulgadas y la versión de Android es 4.0.3, por lo tanto nos vale perfectamente para realizar la pruebas.

Para poder introducir las aplicaciones dentro de nuestro móvil debemos seguir un proceso para que el teléfono nos permita instalar aplicaciones externas. Lo primero que hay que hacer es ir a "Ajustes", entrar en "Opciones de desarrollador" y dentro activar la "Depuración de USB". Esto nos permitirá intercambiar datos entre el ordenador y la memoria interna del teléfono móvil.



Nombre	Fecha de modifica...	Tipo	Tamaño
.settings	29/07/2013 22:43	Carpeta de archivos	
assets	29/07/2013 22:43	Carpeta de archivos	
bin	23/08/2013 17:31	Carpeta de archivos	
gen	06/08/2013 19:10	Carpeta de archivos	
libs	29/07/2013 22:43	Carpeta de archivos	
res	05/08/2013 14:19	Carpeta de archivos	
src	29/07/2013 22:43	Carpeta de archivos	
.classpath	29/07/2013 22:44	Archivo CLASSPA...	1 KB
.project	22/04/2013 12:40	Archivo PROJECT	1 KB
AndroidManifest	16/08/2013 21:44	Documento XML	2 KB
ic_launcher-web	22/04/2013 12:40	Imagen PNG	51 KB
lint	04/07/2013 22:39	Documento XML	1 KB
proguard-project	22/04/2013 12:40	Documento de tex...	1 KB
project.properties	04/07/2013 22:39	Archivo PROPERTI...	1 KB

Figura 65. Carpeta CarruselPFG en la carpeta Workspace.

Lo siguiente será introducir el archivo necesario en nuestro móvil. Para ello debemos buscar la carpeta "Workspace" en nuestro ordenador, que es donde se encuentran los proyectos que creamos con Eclipse. Una vez dentro de la carpeta de nuestro proyecto (Figura 65), buscamos una carpeta con el nombre "bin" y dentro se encontrará un archivo con el nombre de nuestro proyecto y la extensión apk, en nuestro caso "Y si llueve.apk" (Figura 66).

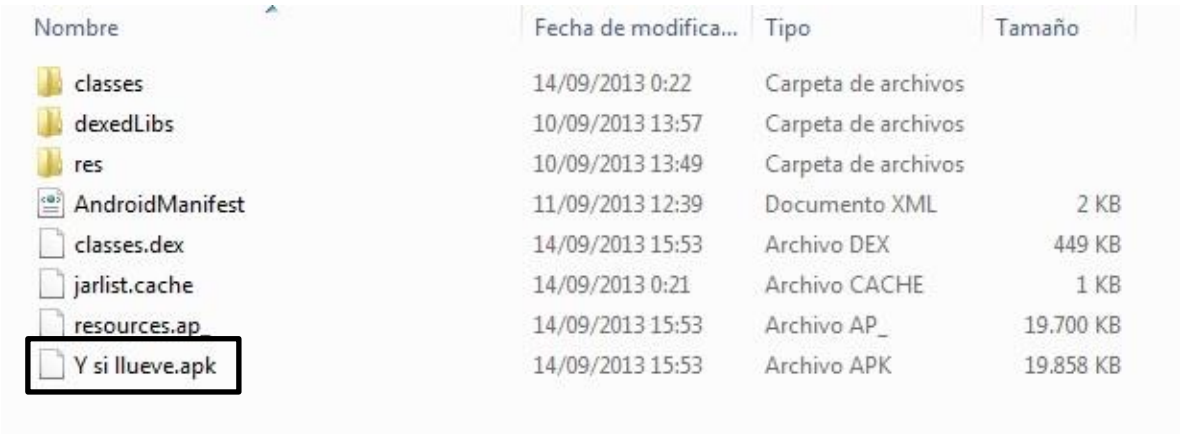


Figura 66. Carpeta bin donde está el archivo de extensión apk.

Una vez que ya hemos hecho esto, debemos conectar nuestro teléfono móvil al ordenador y meter la aplicación dentro. Desconectamos nuestro móvil y ejecutamos el archivo de la aplicación dentro de nuestro móvil. Realizamos la instalación (Figura 67).

Figura 67. Instalación de la aplicación en el móvil.

Hecha la instalación, la aplicación se encontrará en nuestro teléfono móvil. Buscamos el icono correspondiente y lo ejecutamos (Figura 68).

Figura 68. Aplicación funcionando en el teléfono móvil.

4. SUBIDA A GOOGLE PLAY

En este capítulo se cuenta la información principal necesaria por si en un futuro se desease subir la aplicación al mercado de Android.

4.1 REQUISITOS PARA SUBIR UNA APLICACIÓN A GOOGLE PLAY

Para poder subir nuestra aplicación a Google Play es necesario que nos registremos como desarrollador en la página. Los desarrolladores que se registren como nuevos deberán pagar una única cuota de 25 Dólares. Este pago, más que una búsqueda de conseguir dinero, se usa para bloquear el registro y aparición de aplicaciones de baja calidad o productos con Spam. Hay que destacar que, por ejemplo, los desarrolladores de Apple deben pagar 100 Dólares anuales para mantener su licencia de desarrollador [10].

Si el usuario ya posee una cuenta de Google, podrá usarla sin la necesidad de registrarse. En caso contrario deberá crearla para después proceder al pago de la cuota mediante tarjeta de crédito. La dirección donde realizar el registro y el pago es la siguiente: https://play.google.com/apps/publish/signup/. En esa página se pueden consultar los países donde está permitida la distribución y venta de aplicaciones, ver las condiciones de acuerdo de distribución para desarrolladores de Google Play y cambiar la cuenta desde la que quieras pagar.

Hay que tener cuidado al publicar la aplicación ya que Google es muy estricta con el cumplimiento de sus directrices políticas. Algunos ejemplos de motivos para suspender una aplicación son:

- Infracciones por Spam, es decir, publicar contenidos repetidos, engañosos o irrelevantes. También se considera Spam la descripción engañosa de la aplicación para que sea más accesible.

- Infracción de los derechos de propiedad intelectual.

- Inclusión de anuncios que dificultan el uso de la aplicación.

- Publicación de contenido sexual explícito así como imágenes de violencia gratuita.

- Transmisión de elementos dañinos para los dispositivos como son virus, defectos, troyanos o cualquier tipo de software malicioso.

4.2 INFORMACIÓN NECESARIA SOBRE LA APLICACIÓN

Google te exige un cierto número de detalles o información sobre la aplicación para poder subirla a la Play Store (Figura 69). Son los siguientes:

Idioma: Sirve para indicar el idioma de la aplicación. También se pueden incluir traducciones del nombre y de la descripción de la aplicación para así poder llegar a usuarios de otros idiomas.

Nombre: Es el nombre de la aplicación. Se pueden añadir diferentes nombres para cada idioma.

Descripción: La descripción de nuestra aplicación, sus principales funciones y características. El límite máximo de caracteres es de 4.000.

Cambios recientes: Esta parte se utiliza para explicar a los usuarios los cambios que se han ido incluyendo en la aplicación a lo largo de las diferentes actualizaciones que haga el desarrollador.

Texto Promocional: Es el texto situado junto al gráfico promocional en lugares destacados de Google Play.

Tipo de Aplicación: En este caso se ofrecen dos opciones entre las que elegir, Aplicaciones y Juegos.

Categoría: Se deberá elegir una categoría para la aplicación de entre las numerosas que ofrece Google Play. Algunas de ellas son: Comunicaciones, Finanzas, Salud y bienestar, Fotografía, Noticias y revistas, Tiempo, Sociedad, Deportes, etc.

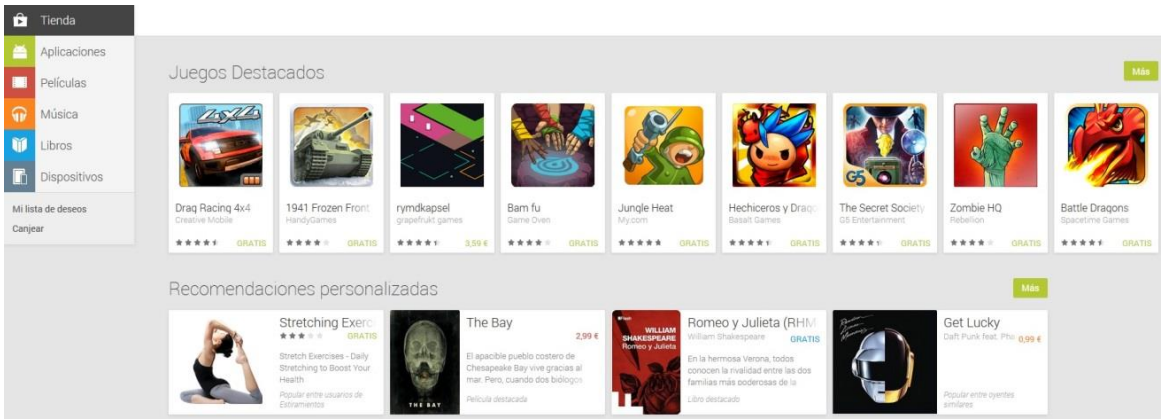


Figura 69. Portada de la página Google Play.

5. CONCLUSIONES Y TRABAJOS FUTUROS.

En este capítulo se muestran las conclusiones tras el desarrollo de la aplicación y los objetivos cumplidos. Además se marcan las pautas para un posible desarrollo futuro de la aplicación con más componentes.

5.1 CONCLUSIONES

Uno de los puntos más importantes a tener en cuenta sobre la realización de este proyecto, es que me enfrentaba a un lenguaje de programación totalmente nuevo y que desconocía, ya que en Electrónica Industrial se enseña a programar en C++, y por lo tanto empezaba de cero.

Una de las cosas que he descubierto es la multitud de información que existe en internet, aunque esto a veces puede ser un inconveniente. Debido a que es un lenguaje de programación relativamente joven, la información a veces es un poco difusa. Sin embargo existen multitud de tutoriales en internet, tanto gratuitos como de pago, con los que cualquier persona puede aprender a hacer una aplicación sencilla. Para poder avanzar más se necesitaría tener más conocimientos de programación y dedicarle muchas horas.

Los objetivos predispuestos al comienzo de la memoria se han cumplido completamente. La aplicación realiza todas las funciones predispuestas de una manera correcta y de fácil manejo para el usuario.

El incremento exponencial de la presencia de la tecnología móvil en nuestras vidas ha abierto un nuevo mercado y son muchas las empresas y desarrolladores que se aventuran en este mundo. Mi experiencia durante el desarrollo de esta aplicación me ha servido para asentar unas bases que pueden serme muy útiles en mi futuro en el caso de que quisiera avanzar en este campo.

5.2 TRABAJOS FUTUROS

Tras el desarrollo completo de los objetivos a continuación se muestran posibles mejoras y avances para futuros trabajos con la aplicación:

- **Traducción del texto de las tarjetas:**

Pese a que nuestra aplicación está disponible en dos idiomas (español e inglés) existe un problema y es que el texto de las tarjetas, al ser imágenes, no se pueden traducir de manera automática. Por lo tanto, una forma de mejorar esta aplicación sería intentando duplicar las tarjetas incluyendo versiones en los dos idiomas. Sin embargo, aunque la aplicación es capaz de reconocer el idioma del dispositivo en el que ha sido instalado, no ocurriría lo mismo con las tarjetas. Esto provocaría que para obtener un

correcto funcionamiento, tendríamos que cambiar parte de la programación de la aplicación o incluir una opción en el menú para traducir.

- **Posibilidad de usar las fotos como fondo:**

Una opción que sería muy interesante para los usuarios de la aplicación sería el poder usar las imágenes que aparecen en el carrusel como fondo para su propio teléfono móvil. Esto daría valor a la aplicación y al estar visible fomentaría la aparición de futuros usuarios que pudieran interesarse por el fondo del teléfono de un conocido.

- **Inclusión de música:**

Durante el desarrollo de la aplicación se barajó la posibilidad de incluir una música de fondo que sonase mientras estuviese la aplicación en funcionamiento, o sólo cuando se ejecutase el carrusel. Sin embargo esto se ha desestimado por un motivo principalmente. Hoy en día los Smartphones se usan en muchas ocasiones como reproductores de música. Si se ejecutase la aplicación mientras se está escuchando alguna canción, ambas se solaparían resultando desagradable para el usuario. Este problema se puede evitar programando la aplicación para que detecte si hay un audio ya reproduciéndose o provocando que la propia aplicación silencie el otro audio. Debido a la complejidad de estas opciones se deja para un desarrollo futuro.

- **Añadir publicidad:**

En la Play Store de Android podemos encontrar aplicaciones tanto gratuitas como de pago. Las aplicaciones sin coste para el usuario suelen contener publicidad añadida que sirven para financiar futuros proyectos del desarrollador. Además en muchas ocasiones se ofrece una versión de pago de una aplicación gratuita en la que simplemente se quita la publicidad. En nuestro caso, al ser un proyecto universitario la aplicación es gratuita y no contiene publicidad, sin embargo en un futuro se podría buscar obtener algún beneficio incluyéndola.

La mayoría de la publicidad usada en aplicaciones es del tipo que se muestra en la Figura 70, es decir, una línea pequeña en la parte inferior o superior de la aplicación y que no incomoda el correcto funcionamiento de la aplicación.

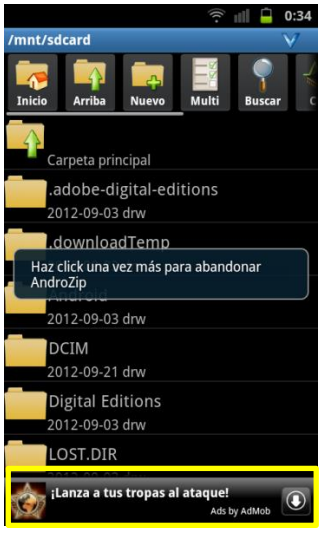


Figura 70. Ejemplo de publicidad incluida dentro de una aplicación.

6 PRESUPUESTO

En este apartado se va a detallar el presupuesto del proyecto. Para calcular los costes hemos tenido en cuenta las siguientes consideraciones:

- El inicio del proyecto se sitúa en el 1 de mayo de 2013 y la fecha de fin el 2 de septiembre de 2013.
- Se estiman 5 horas trabajadas por día laborable.
- Se cuentan como no laborables los fines de semana y se añaden como festivos los correspondientes a esas fechas en la Comunidad de Madrid y que hacen un total de tres días más no laborables. El número total de días trabajados son 77.
- En total el número de tiempo empleado en el proyecto es de 385 horas.
- Se recuerda que tanto el SDK de Android como el programa Eclipse son totalmente gratuitos y están a la disposición de cualquier persona.

Coste de personal:

A continuación se detallas los costes asignados al personal en el proyecto. La única persona que ha participado en la realización del proyecto ha sido el autor de la memoria y por lo tanto ha sido el encargado de la realización de todas las tareas expuestas (Tabla 1).

Empleado	Coste (€/Hora)	Número de horas	Coste total (€)
Analista	20 €/hora	85	1700€
Diseñador	18 €/hora	100	1800€
Programador	15 €/hora	200	3000€
TOTAL		385	6500€

Tabla 1. Costes de personal.

Coste de hardware:

En este apartado se detallan los gastos del proyecto asignados a los elementos del hardware que han sido imprescindibles para el desarrollo de la aplicación.

Concepto	Coste (€)	Tiempo de uso (meses)	Vida útil (meses)	Coste (€)
Portátil Asus K53SD	585€	4	60	39
Samsung Galaxy SII	435€	4	36	48,34
TOTAL				87,34

Tabla 2. Costes de hardware.

La fórmula utilizada para calcular los costes del hardware es la siguiente:

$$Coste = \frac{A}{B} \times C$$

Donde:

A= Número de meses en el que se utiliza el equipo.

B = Vida útil del equipo utilizado.

C= Coste del equipo.

Coste total:

CONCEPTO	PRECIO (€)
Coste de personal	6500
Coste de hardware	87,34
IVA (21%)	1383,34
TOTAL	**7970,68**

Tabla 3. Costes totales.

7 BIBLIOGRAFÍA

- [1] **http://techland.time.com/** : Página sobre tecnología, usada para los gráficos sobre las comparativas de sistemas operativos. – *Última visita: 15/08/2013.*

- [2] **http://developer.android.com/index.html** : Página oficial para los desarrolladores de Android. – *Última visita: 27/08/2013.*

- [3] **http://developer.android.com/sdk/index.html** : Página donde descargar directamente el SDK de Android. – *Última visita: 10/05/2013.*

- [4] **Pragmatic – Hello Android, 3rd Edition 2010.** – *Última consulta: 12/08/2013.*

- [5] **http://www.androidcurso.com/ :** Curso online sobre Android y su programación. – *Última visita: 26/08/2013.*

- [6] **http://www.marinaanaya.com/es/y-si-llueve:** Página web de la autora de los diseños. – *Última visita: 28/08/2013.*

- [7] **https://www.elbauldelprogramador.com:** Página web con información y utilidades para programadores. – *Última visita: 02/09/2013.*

- [8] **http://androideity.com/ :** Página sobre Android, con herramientas, cursos y guías. – *Última visita: 02/09/2013.*

- [9] **http://www.inter-fuser.com/2009/08/android-animations-3d-flip.html:** Página en la que se explica el uso del efecto en tres dimensiones usado en nuestra aplicación. – *Última visita: 04/07/2013.*

- [10] **https://play.google.com/ :** Página oficial de Google Play, lugar donde se alojan todas las aplicaciones disponibles para Android. – *Última visita: 02/09/2013.*